CURRICULUM OVERVIEW

DESCRIPTION

This is the first of two books containing the Atari Computer Camp BASIC curriculum. The lessons were designed to cover the needs of campers with a wide range of abilities and computing background. It is most likely that the group using this book will be made up of individuals who have had some experience with computers. They might have had an introductory course with some programming in BASIC, PILOT or Logo. They should have typed in programs, and they should be able to make modifications to existing programs. It might be necessary to have older campers begin in this book in order to place them with the appropriate age group. If that is the case, they will need additional assistance until they catch up with the group.

The book is divided into three sections. The first contains this overview, instructions about activities that should take place the first days of camp and about non-curriculum related activities such as using Bank Street Writer, Home File Manager, Paint, and other software. The second part is made up of seven curriculum modules, and the third is an appendix with program listings, charts, and other general information.

The seven modules cover the following topics:

Review of Simple Programming and Atari Editing
Atari Sound and Graphics
Variables
Loops
Subroutines and Structured Programming
Conditionals
Project Using Utility Programs.

Each module has lessons whose directions are specific enough to make it possible for a Teaching Assistant to instruct a small group of campers if necessary, while the teacher instructs the majority of the group or works with individuals who are completing projects from a prior session.

THE BOOK IS NOT MEANT TO BE USED IN ITS ENTIRETY! Many activities can be done without having completed work in previous modules, although some lessons require prior knowledge of a concept. For example, campers should know how to use variables before they begin the module on loops. It would be best if you became familiar with the content of each module, so that you can choose tasks appropriate for each group or individual you are teaching.

To help you find different types of activities, a code is placed next to the page number at the bottom of each page.

LP - A Lesson Plan page. This could be directions for the teacher or teaching assistant, or a student worksheet necessary for the lesson.

IA - Individual Activity. An activity for a camper who needs review or practice, but does not need a formal lesson.

T - Indicates that this page is also available as a Transparency.

C - Chart. These are pages that will be useful for more than the lesson in which they are introduced. Charts have been duplicated in quantity, so that campers can keep them in their binders for reference.

MATERIALS

The list below includes all materials necessary to teach the lessons in this book. Not all lessons require all of the items. Refer to the specific module's cover sheets to identify materials needed for that module.

BASIC Cartridge
BASIC Utility Disk
Personal Diskette for each camper. This is a blank diskette that the camper will use to store programs.
Atari Graphics Modes Paper
Control Graphics cards or duplicated sheets
General Utility Diskette
PAINT and PAINT Data Diskette
Player Maker
BASIC Reference Booklet (One per machine)

The following optional support materials will also available soon after camp starts.

Duplicated copies of selected articles from the Atari Connection. These are available in classroom sets in the library. Some will have cross references in the curriculum itself.

A list of one minute "filler" activities that includes such things as famous names in computing, the history of computers, word games, jokes, vocabulary building, etc.

RESOURCES

The following books are in the library and have programs or information that will be helpful.

Inside Atari Basic

Your Atari Computer

Atari Sound and Graphics

101 Atari Programming Tips and Tricks

Atari BASIC - Learning By Using

Making of a Micro

Armchair BASIC

Atari Games and Recreation

You Just Bought a Personal What?

CURRICULUM EVALUATION

It would be best if you documented the way you used this book and the degree of success you encountered in each lesson as you teach the lessons. Make notes on the book itself. The books will be collected and the notes used to help evaluate each module. Put a tally mark of some sort on each page you use each time you use it.

Use the computer network to send lists of pages that contain errors such as misspelled words, errors in programs, etc. We will be on the network every day, so it will be much easier and faster to use that media rather than receiving information by U. S. Mail. It is especially important that any student worksheets that have errors on them be reported immediately, so that we can send corrected copies to replace them. Do not send pages home that have errors on them. Remove them from the campers' binders and we will replace them (if time permits) before they leave camp. For mistakes in the teacher's binders, you will be informed (using the network) that appropriate changes need to be made, and you will be asked to make them in pen on your copy of the curriculum.

CONTENT SUMMARY

The content of each part of the book is summarized below. For more detailed information about objectives and activities in the module see the individual module cover sheet.

INTRODUCTION

Curriculum Overview
Getting Started - The first days of camp.
Software campers are required to use
"Personal" Documentation - The "getting to know you" program

MODULE #1 - Review of Simple Programming, Atari Editing

Review of edit features, control graphics and simple BASIC programming. Lessons include:

Editing Text
Immediate Mode and Control Graphics
Editing Programs
Operators
LPRINT
POSITION

Should be used to help campers with some experience learn to use the Atari keyboard and editing features.

MODULE #2 - Graphics and Sound

Introduction to graphics and sound capabilities of the Atari. Lessons include:

GRAPHICS 0,1,2
GRAPHICS 3 - 8
SETCOLOR
COLOR
SOUND

It is important that campers be familiar with the content of each of these lessons, since later programming builds on what is presented in this module.

MODULE #3 - Variables

Using numeric and string variables in programs. Stresses understanding the concept of a variable. Lessons include:

Variables
LET
INPUT
RND & INT

The RND and INT functions are used in subsequent modules, thus it is important to do the activities in that lesson.

MODULE #4 - Loops

In order to complete the activities in this module, campers must be familiar with variables. Lessons include:

FOR..NEXT
FOR..NEXT..STEP
NESTED LOOPS
GOTO

The activity in the GOTO lesson would be of interest to most campers even though they might already understand GOTO.

MODULE #5 - Subroutines and Structured Programming

Campers use subroutines to produce a story containing text, graphics, and sound effects. Structured programming is emphasized. Lessons include:

Subroutines, An Introduction
Practice Using Subroutines
Graphics Subroutines
Sound Subroutines
Writing a Program

It is hoped that as many campers as possible will finish the curriculum at least through this module. Read it thoroughly and note what needs to be taught before you begin. Then use that information to select and prioritize activities from the previous modules. (In some cases, it will not be possible for campers to finish the module. Use your discretion in pacing with different individuals.)

MODULE #6 - Conditionals

Covers the use of conditionals for program control and branching. Campers must be familiar with variables and looping before beginning this module. Lessons include:

IF..THEN with Numeric variables
IF..THEN with Strings
OR and AND
Using a Joystick
Paddle Controllers

The IF..THEN with strings includes a program that deals with famous names in computing.

MODULE #7 - Individual Project

Campers learn to produce a "computer show" with animation using PAINT, PLAYER MAKER, and utility programs on the BASIC UTILITY DISKETTE. It takes a long time to finish the activities in the module. Some knowledge of BASIC is necessary, but with direction, campers could begin work on very early in the session.

APPENDICES

APPENDIX A - Programs on the BASIC Utility Diskette

APPENDIX B - Error messages

APPENDIX C - Basic reserved words

APPENDIX D - ATASCII Character Set

Getting Started

For those campers who have never used an Atari computer, or maybe never used a disk drive, you will need to present a basic introduction to the handling of equipment and diskettes. A review of this topic may be in order for most everyone. Once this is done, any of the following activities may be done at the beginning of camp in whatever order you like. (Also see BASIC curriculum, module 1, and PILOT curriculum.)

1. Campers should take turns playing one of the typing games available at camp. Both MASTER TYPE and TYPO ATTACK were sent in large quantities. This activity does not require an instructor's presence, but will require an introduction, and thus you may wish to use it as a "side attraction" while other activities are in progress.

2. Get the campers to run the PERSONAL program so that they might get acquainted with each other. This program can be found on the BASIC UTILITY DISKETTE. Instructions are included in this packet.

3. Teach campers (those who don't already know) how to format and write DOS to a blank diskette. They will need to do this before you can continue with the curriculum.

4. Show campers the library setup and explain the checkout system. Pass out the library cards. Stuff binders.

5. Give campers an early initial experience with the BANK STREET WRITER word processor so that they can begin to write letters home.

Other software that campers should get exposed to sometime during their stay at camp (during the teaching assistant workshops on Tuesdays) include the following:

PAINT
Home Filing Manager
Factory
Odell Lake (Elementary Biology)
Magic Melody Box
An Adventure game

Other items of interest for campers to be exposed to include:

Topo Robot
Alien Voice Synthesizer
Four Color Plotter
The camera for taking screen pictures.

INSTRUCTORS' INSTRUCTIONS FOR "PERSONAL"

Personal is actually three programs in one. Instructors will only use the second of these three to dump the campers data onto the main data disk and to load a random record for the MYSTERY PERSON program. Campers will use the other two to enter data and to search for their random person off of the main data disk.

RUNNING THE PERSONAL PROGRAM

1. Make sure the campers' computers and disk drives are all turned off.

2. Have them insert the ATARI BASIC Language Cartridge into the left-hand cartridge slot in the computer.

3. Campers should now turn on their disk drives. When the busy light goes out, have them open the disk drive door and insert the program diskette, with the label in the lower right-hand corner nearest them.

4. Have the campers turn their computers and T.V. sets on.

5. When the READY prompt appears on their T.V. screens, the campers should type RUN "D:PERSONAL" and press the RETURN key. The program will then load into computer memory and start.

6. The campers can now use the "PERSONAL" program by answering all of the questions with the appropriate answer, until the screen that says "THE END" appears.

USING THE DATA DUMP ROUTINE

1. After the campers have finished entering their data they will be instructed by the program to report to the instructor for an introduction to some new piece of software or hardware.

2. All of the computer screens should now read "THE END".

3. To start dumping the data for the room onto the main data disk, press the control and 's' keys at the same time.

4. You will be prompted to insert the data disk and press return twice. The program will then dump the campers data to the main data disk.

5. When "Finished" appears on the screen and the busy light

goes out remove the data disk from the drive.

6. Repeat steps 3,4, and 5 until finished with the room.

7. When you are finished with the room, press the control and 'e' keys at the same time. You will be prompted to insert the data disk and press RETURN twice.

8. Repeat step 7 until finished with the room.

USING MYSTERY PERSON

1. The first thing the camper should see is the MYSTERY PERSON title followed by the words "PRESS START". To continue form here the camper must press the start button.

2. The MYSTERY PERSON program will then begin by giving each camper his/her first clue.

3. When they think they have guessed the mystery person the camper should enter the mystery persons name and type RETURN.

4. At this point the program will either let the next camper guess who their mystery person is or it will congratulate them on their correct guess.

5. If after ten clues the campers have not guessed who thier mystery person is, the program will print thier name followed by their suspect's name. After a short time a new screen saying MYSTERY PERSON will appear.

6. Because some campers may finish before others, you may wish to have them play MYSTERY PERSON again by first inserting the data disk, then by pressing the control and 'c' keys at the same time. The computer will choose another mystery person off of the data disk.

7. To stop the program just have the campers turn the computers off.

HOW TO USE DATABASE

INTRODUCTION

DATABASE is a program that allows you to search the main disk database that was created with the PERSONAL program. With DATABASE you can either search the disks database or you can list it.

SEARCH DATABASE

1. Select SEARCH DATABASE from the MAIN MENU.

2. You should now be in the SEARCH DATABASE menu.

3. Position the cursor with the arrow keys, next to the headings you would like to search through.

4. Now type the asterisk '*' to tell the computer that this is one of the headings that is to search for.

5. Repeat steps 3 and 4 until you have finished making all of your selections.

6. At this point you may either continue by pressing the START button, or return to the MAIN MENU by pressing OPTION.

7. If you press START, the computer will ask you to enter the headings it is to search for, and then to type return.

8. After you type return you will be asked if you want the computer to search for your heading or everything but your heading. What this means is if you choose to have the computer search for your heading it will do just that, search for your heading. If you choose for the computer to search for everything but your heading, it will search through the database and only look for the information that doesn't match your heading.

9. The computer will repeat step 8 until you have entered all the information for the headings that you selected from steps 3 and 4.

10. The last menu will ask you if you would like to have the people who fit your description, printed out on the printer. Choose either print or don't print with the arrow keys and then press the START button to continue.

11. You should now see your description followed by the people who match it from the database.

LIST DATABASE

1. Select LIST DATABASE from the MAIN MENU.

2. You will see a menu that looks similar to the menu for SEARCH DATABASE. The difference between the two is that this menu only allows you to make one heading choice.

3. Choose one of the headings. This heading will be what the computer will search through later in the program.

4. The next screen lets you choose between one of the following: 1. Having the computer search the database for a single letter/number. 2. Having the computer search the database between two letters/numbers.

5. Enter the letters or numbers in the next section. If you want to erase your entry press the space-bar.

6. Next the computer will ask you if you want the information printed out on the printer. You may make your choice by first moving the cursor with the arrow keys then by pressing the start button to continue.

7. The computer will now search the database through the heading you chose in step 3. It will look for a match from the entry you made in step 5. If it finds a match it will list that person's information on either the printer, the screen, or both, depending on what you chose in step 6.

HOW TO USE DATALINK

DATALINK is a program that was created to join the many classroom database's into one main camp database.

1. To get started you will need two drives, the disks with the classroom database on them, and the DATABASE/DATALINK program disk which will be used to store the main database.

2. To run the DATALINK program, insert the DATABASE/DATALINK disk into drive #1. Now type RUN "D:DATALINK" then RETURN.

3. Remove the DATABASE/DATALINK disk from the disk drive and place the classroom database disk into drive #1, then press return. If the file CAMPER.DAT isn't on the disk the prompt "Insert the classroom database into drive #1" will repeat.

4. Now put the DATABASE/DATALINK disk that will hold the main database into drive #2 and press return. If there isn't a CAMPER.DAT file on this disk, the program will create its own.

5. The program will now transfer the CAMPER.DAT file from drive #1 to drive #2.

6. Next the prompt "Would you like to add another classrooms data (Y/N)" will appear. If you wish to stop now type 'N'. If you have more data to store in the main database then type 'Y'. The program will then repeat steps 2 through 5 until you have finished.

DATABASE STRUCTURE

The following is a brief explanation on how the database, "CAMPER.DAT" is structured.

CAMPER.DAT is a simple database. It is not alphabetized or indexed. It contains the information which was entered with the PERSONAL program in the same format. What this means is that the information for each person on the disk is arranged in the same way as it was entered. With the persons name first followed by there address, city, state, etc... This continues until you run out of data.

Database format

Name
Address
City
State
Zip
Age
Hair color
Eye color
Sex
Favorite color
Favorite singing group
Favorite song
Favorite movie
Favorite T.V. show
Favorite food
Favorite sport
Favorite game
Favorite animal

The following is a program that will allow you to list the database.

```
10 REM ***LIST DATABASE***
20 DIM TEMP$(30):REM Each record is 30 characters long
30 OPEN #1,4,0,"D:CAMPER.DAT":REM Open file for input
40 TRAP 80
45 REM Print database
50 INPUT #1;TEMP$:REM Input record
60 PRINT TEMP$:REM Print record
70 GOTO 50:REM Repeat until end of file
80 CLOSE #1:END:REM Close file
```

CAMPER'S INSTRUCTIONS FOR "PERSONAL"

RUNNING THE PERSONAL PROGRAM

1. Make sure your computer and disk drive are both turned off.

2. Insert the ATARI BASIC Language Cartridge into the left-hand cartridge slot in your computer.

3. Turn on your disk drive.

4. When the busy light goes out, open the disk drive door and insert the program diskette with the label in the lower right-hand corner nearest to you. Close the door.

5. Turn on your computer and T.V. set.

6. When the READY prompt displays on your T.V. screen, type RUN"D:PERSONAL" then press the RETURN key. The program will then load into computer memory and start.

7. Answer all questions with the appropriate answer.

8. When you have reached the screen that says, "THE END", leave the computer on and report to your instructor.

USING MYSTERY PERSON

1. The first thing you should see is the MYSTERY PERSON title followed by the words "PRESS START". To continue from here press the start button.

2. The MYSTERY PERSON program will then begin by giving you the first clue.

3. When you think you have guessed your mystery person enter their name and type RETURN.

4. At this point the program will either let the next camper guess who their mystery person is or it will congratulate you on your correct guess.

5. If after ten clues you have not guessed who your mystery person is, the program will end by printing your name followed by your suspect's name.

INSTRUCTOR'S INSTRUCTIONS FOR "PERSONAL"

Personal is actually three programs in one. Instructors will only use the second of these three to dump the campers data onto the main data disk and to load a random record for the MYSTERY PERSON program. Campers will use the other two to enter data and to search for their random person off of the main data disk.

USING THE DATA DUMP ROUTINE

1. The campers will first load and run the personal program.

2. After the campers have finished entering their data they will be instructed by the program to report to the instructor for an introduction to some new piece of software or hardware.

3. All of the computer screens should now read "THE END".

4. To start dumping the data for the room onto the main data disk, press the START and SELECT buttons at the same time.

5. You will be prompted to insert the data disk and press return twice. The program will then dump the campers data to the main data disk.

6. When "Finished" appears on the screen and the busy light goes out remove the data disk from the drive.

7. Repeat steps 4,5 and 6 until finished with the room.

8. When you are finished with the room, reinsert the data disk into each drive and press return twice. This will randomly pick one of the records off of the main data disk for the campers to use with the MYSTERY PERSON program.

9. Repeat step 8 until finished with the room.

MODULE #1 - REVIEW OF EDIT FEATURES, CONTROL GRAPHICS, AND SIMPLE BASIC PROGRAMMING

OBJECTIVE

To review the following topics:

Editing Text
Immediate Mode and Control Graphics
Editing Programs
Simple BASIC Programming

MATERIALS REQUIRED

BASIC Cartridge
BASIC Utility Disk: PRINTS
Formatted Disk for each camper (This disk should also have DOS files.)
Graph Paper
Control Graphics Cards (or duplicated sheets)

REFERENCES

Inside Atari Basic, pp. 8-39

Your Atari Computer, pp. 20-25, 38, 42-58

Atari 400/800 Basic Reference Manual, pp. 5-6, 9-10, 13-14

CONTENT

The module is divided into seven lessons. Be selective in your use of the activities. There are many more than can be done in the time that you have available.

Lesson 1 - Editing Text

Pages 1-7

Covers special keys on the Atari keyboard. Short selections to copy and edit. Includes a chart of keyboard editing features.

Lesson 2 - Immediate Mode and Control Graphics

Pages 8-13

New Statements
PRINT

Materials
BASIC Cartridge

Familiarizes students with control graphics characters, printing strings, and numeric operations and precedence

Lesson 3 - Editing Programs

Pages 14-19

New Statements
NEW RUN SAVE LOAD LIST

Materials
BASIC Utility Disk
BASIC Cartridge
Graph Paper

Allows students to practice using editing features covered in previous lessons. Reviews use of commands.

Lesson 4 - Editing Programs-Part II

Pages 20-24

New Statements
REM END

Materials
BASIC Cartridge

Reviews simple BASIC programs. Editing lines, changing line numbers, number of screen lines for each print statement, listing parts of programs.

Lesson 5 - Operators

Pages 25-27

Materials
BASIC Cartridge

Reviews use of semicolon, comma, and colon in BASIC programming.

Lesson 6 - LPRINT

Pages 28-30

New Statement
LPRINT

Materials
BASIC Cartridge

Reviews use of LPRINT.

Lesson 7 - POSITION

Pages 31-32

New Statement
POSITION (POS.)

Materials
BASIC Cartridge
Graph Paper

Reviews use of POSITION in Graphics 0.

BE SELECTIVE. THIS MODULE IS NOT MEANT TO BE USED IN ITS ENTIRETY! It would be best if you became familiar with its parts, so that you can choose tasks appropriate for your group or individuals.

Each lesson contains several types of material. Next to the page number at the bottom of each page is a code which is designed to help you find appropriate material quickly.

LP	A Lesson Plan page. This could be directions for the teacher or aide, or a student worksheet necessary for a lesson.
IA	Individual Activity. An activity for a student who needs review or practice, but does not need a formal lesson.
T	Indicates that this page is also available as a Transparency.
C	Chart. These are pages that will be useful for more than the lesson in which they are introduced.

CONTENT

The module is divided into seven lessons. Because it is so long, the content of each lesson is dealt with separately.

Lesson 1 - Editing Text

Pages 1-7

Covers special keys on the Atari keyboard. Short selections to copy and edit. Includes a chart of keyboard editing features.

Lesson 2 - Immediate Mode and Control Graphics

Pages 8-13

New Statements
PRINT

Materials
BASIC Cartridge

Familiarizes students with control graphics characters, printing strings, and numeric operations and precedence.

<u>Lesson 3</u> - <u>Editing Programs</u>

Pages 14-19

New Statements
NEW RUN SAVE LOAD LIST

Materials
BASIC Utility Disk
BASIC Cartridge
Graph Paper

Allows students to practice using editing features covered in previous lessons. Reviews use of commands and how to add and delete lines from programs. Emphasizes necessity to press return after editing a line in a program.

<u>Lesson 4</u> - <u>Editing Programs-Part II</u>

Pages 20-24

New Statements
REM END

Materials
BASIC Cartridge

Reviews simple BASIC programs. Editing lines, changing line numbers, number of screen lines for each print statement, listing parts of programs.

<u>Lesson 5</u> - <u>Operators</u>

Pages 25-27

Materials
BASIC Cartridge

Reviews use of ":", ";", and "," in BASIC programming.

Lesson 6 - LPRINT

Pages 28-30

New Statement
LPRINT

Materials
BASIC Cartridge

Reviews use of LPRINT.

Lesson 7 - POSITION

Pages 31-32

New Statement
POSITION (POS.)

Materials
BASIC Cartridge
Graph Paper

Reviews use of POSITION in Graphics 0.

EDITING TEXT

An individual worksheet called "PRACTICE EDITING TEXT" and a correction guide are available for campers who need practice, but do not need a formal lesson.

Activity #1

If campers are not familiar with the Atari keyboard, they should be introduced to the following editing features before attempting this exercise.

Caps/Lowr
Ctrl- (with arrows)
Delete Back S
Shift-Delete Back S
Ctrl-Delete Back S
Ctrl-Insert
Automatic repeat feature
Inverse video
Zero vs. "O"
Ctrl-Clear

A summary of the keystrokes called "KEYBOARD EDITING FEATURES" is included with the worksheets. It is designed to be used as a reference for campers who have some knowledge of Atari text editing, so that they can complete the exercise on their own if necessary. It may also be used by beginners after an initial lesson has been presented by the teacher or aide.

Activity #2

Be sure campers have removed the Atari Basic cartridge and are in the Atari Memo Pad mode. They should copy the first paragraph from the student worksheet using the memo pad. Instruct campers to copy the paragraph exactly as it is written, even though there are mistakes. Since the purpose of the exercise is to practice editing text, they need not correct typing errors until they enter the editing stage. The intentional mistakes in the text are underlined. They are given the paragraph in the correct form to use when editing. The idea is to use as few keystrokes as possible to make the corrections. It would be most beneficial if you guided them through the actual editing. You might want to use the "CORRECTION GUIDE" that accompanies the individual worksheet for ideas about how to approach instructions for corrections.

A transparency and individual camper copies of the text (called "PRACTICE EDITING") are available for this exercise.

EDITING PRACTICE

PARAGRAPH #1

First identified by indian tribes more than a centry ago, Bigfoot has bafffled thousands of investigators, who have attampted to track him down. Many scientists, in fact, skepticla about the animal's existence, arguing that if an American Primate existed, a live specimen would have been discovered discovered by now.

****
Note that the word, "remain", must be inserted as part of the exercise.
****

CORRECT PARAGRAPH

First identified by Indian tribes more than a century ago, Bigfoot has baffled thousands of investigators, who have attempted to track him down. Many scientists, in fact, remain skeptical about the animal's existence, arguing that if an American primate existed, a live specimen would have been discovered by now.

PRACTICE EDITING TEXT

Be sure you are in the Atari Memo Pad Mode. That can be accomplished by removing the BASIC cartridge or by typing "BYE" if you are in BASIC. Copy the following text exactly as it is written. You will make corrections after you have finished typing the whole section. The letters and words that are underlined show you what changes need to be made. You will type them as regular letters or words, without the underlining. The '*' shows were a word has been omitted. You should leave out the '*' when you make corrections.

Color Register--The zpecific
location in and computers memory
that stores the colro you tell it to.
pEEK--A BASIC comnand that tells
the computer to look into a specific
location in the computer''s memory
and see what what is stored there.
POKE--A * command that tells
the computer to put a new number into
a specific location in the computer's
memory.

Now you are ready to correct the paragraph. Use the guide on the next page to make the corrections. The idea is to help you learn to use as few keystrokes as possible, so you should not retype whole lines to correct mistakes. The corrected version below should help you to check your final copy.

CORRECTED VERSION

Color Register--The specific
location in the computer's memory
that stores the color you tell it to.
PEEK--A BASIC command that tells
the computer to look into a specific
location in the computer's memory
and see what is stored there.
POKE--A BASIC command that tells
the computer to put a new number into
a specific location in the computer's
memory.

CORRECTION GUIDE
FOR "PRACTICE EDITING TEXT"

- <u>z</u>pecific
  Position the cursor over the z. Type an "s".

- <u>and</u>
  Position the cursor over the a. Type "the".

- computer<u>s</u>
  Position the cursor over the s. Use CTRL-INSERT to get a space. Type SHIFT-7 to get the "'" mark.

- col<u>ro</u>
  Position the cursor over the r. Type "or".

- <u>p</u>EEK
  Position the cursor over the p. Press SHIFT-p to get a capital P.

- com<u>n</u>and
  Position the cursor over the n. Type an "m".

- computer<u>'</u>'s
  Position the cursor over the "'" that is underlined. Press CTRL-DELETE BACK S.

- <u>what</u>
  Position the cursor to the right of the t. Press CTRL-DELETE BACK S until the whole word has been erased.

AT THIS POINT THE OMITTED WORD SHOULD BE INSERTED.
Position the cursor over the "c" in the word "command". Press CTRL-INSERT one time for each letter and space to be inserted (6). Type the word, "BASIC ". Be sure to leave a space after the word.

KEYBOARD EDITING FEATURES

SELECTED SINGLE KEYPRESSES

KEYSTROKE	CHARACTER OR ACTION
SYSTEM RESET	Stops everything. USE CAUTION!
RETURN	Signals line is finished
BREAK	Interrupts and halts. USE CAUTION!
CAPS/LOWR	Changes to lower case letters
THE /|\ KEY	Inverse video switch
TAB	Moves to next tab stop
DELETE BACK S	Character left of cursor erased, cursor backs up one space

SELECTED SHIFT KEY EFFECTS

KEYSTROKE	CHARACTER OR ACTION
SHIFT-TAB	Set tab stop
SHIFT-Clear	Clear display screen
SHIFT-Insert	Insert blank line
SHIFT-CAPS/LOWR	Switch keyboard to upper-case
SHIFT-DELETE BACK S	Delete current line

SELECTED CTRL KEY COMBINATIONS

KEYSTROKE	CHARACTER OR ACTION
CTRL-TAB	Clear tab stop
CTRL- ↑	Cursor up a line
CTRL- ↓	Cursor down a line
CTRL- ←	Cursor left one space
CTRL- →	Cursor right one space
CTRL-1	Stop/start printing on screen
CTRL-Clear	Clear screen
CTRL-Insert	Insert a space
CTRL-CAPS/LOWR	Switch keyboard to graphics mode
CTRL-DELETE BACK S	Delete character under cursor

IMMEDIATE MODE AND CONTROL GRAPHICS

These activities should be a review for most of the campers in the group. After evaluating the level of your students, you might decide that a group discussion/review is not necessary. In that case, the worksheets may be used for individual activities, or eliminated if appropriate. Duplicates of the control graphics keyboard are available for this lesson.

<u>Activity #1</u> (Do this activity in the Memo Pad mode.)

Use the control key with the appropriate graphics keys to review the symbols. Remind campers that the CTRL-CAPS/LOWR combination locks the keyboard into the graphics character mode. Encourage as much exploration as possible, so that a familiarity with the symbols has been developed. The control graphics will be used in later programming activities. Duplicates of the control graphics keyboard are available for this lesson.

Campers should complete at least the following explorations:

1. Find each of the keys that is used for control graphics.
2. Make a repeated pattern using a combination of two or more keys.
3. Make a picture using any combination of keys.

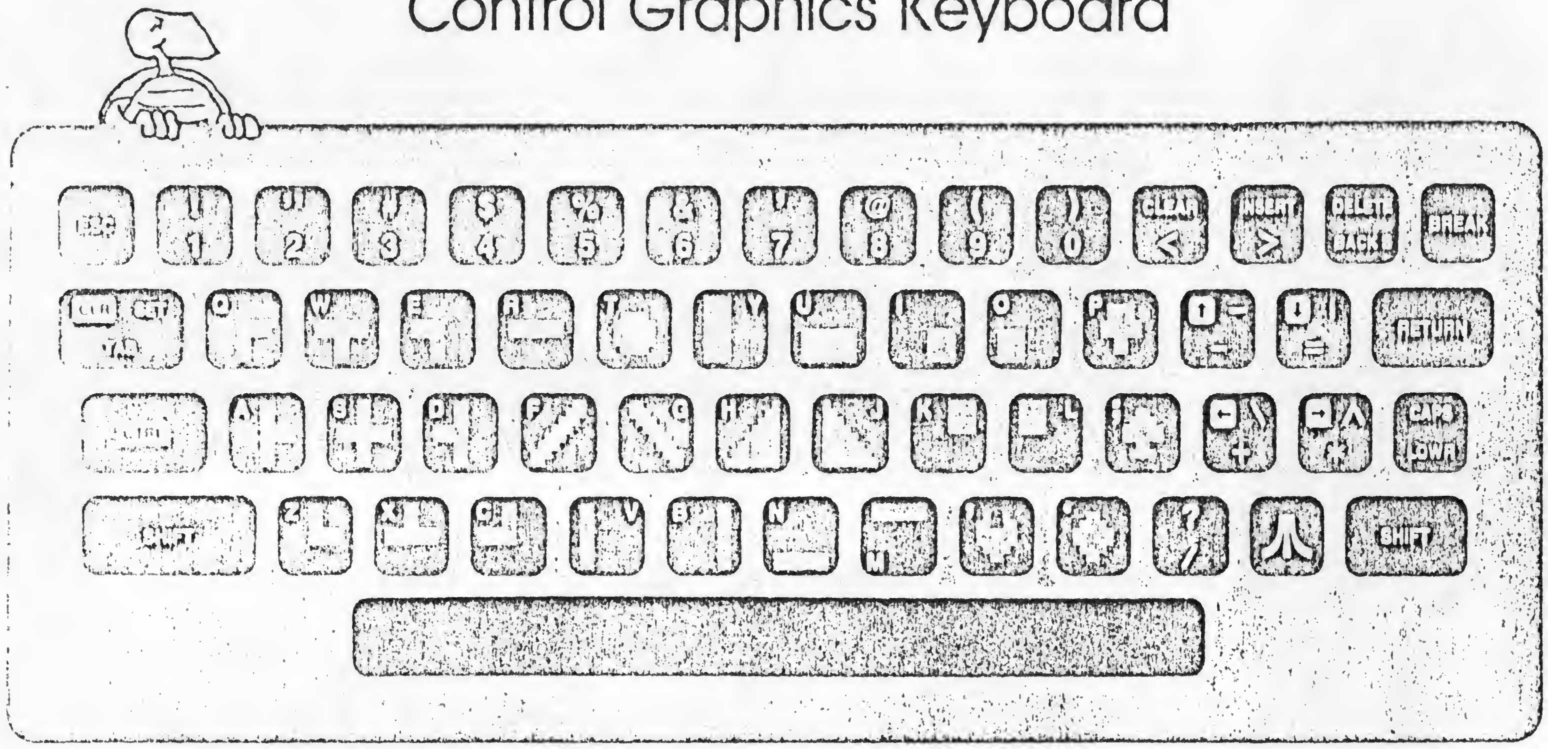
Two worksheets ("EXPLORE CONTROL GRAPHICS" and "CREATE WITH CONTROL GRAPHICS") are available for those campers who have experimented with control graphics. Both are designed to be used as individual activities.

<u>Activity #2</u>

<u>Be sure the Atari BASIC cartridges are in place before you begin this activity.</u>

Campers should use the computer to try each of the print statements on the page called "PRINT STATEMENTS". A transparency is available, as are printed copies. It is important that this be a directed lesson, since campers might miss the point of doing the exercise if they merely type in the information. The items on the list are chosen to provide a review of printing characters and of precedence in arithmetic operations. It may be necessary to do more than one example to illustrate a point.

Control Graphics Keyboard



PRINT STATEMENTS

```
PRINT "4 + 4"

PRINT "4+4"

PRINT 4+4

PRINT 4-4

PRINT 4*4

PRINT 4/4

PRINT 4+3*2

PRINT 4+(3*2)

PRINT (4+3)*2

PRINT 24/2+6

PRINT 24/(2+6)

PRINT 24-8+3-16

PRINT 3+16-70

PRINT 12/4*6-3*5+2

PRINT 12/4*(6-3)*(5+2)

PRINT (2+4+5)*(12/4)+1

PRINT "I'M GREAT!"
```

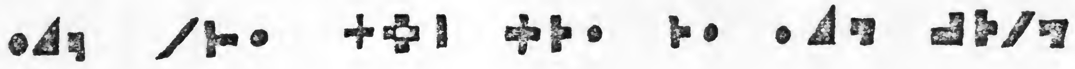
EXPLORE CONTROL GRAPHICS

Directions

1. Be sure you are in the memo pad mode before you begin.
2. Lock the keyboard in the graphics character mode by using the CTRL-CAPS/LOWR combination. In order to use the characters on the ",", ".", and ";" keys, you must use the CTRL key. The remainder of the keys will automatically put the graphics characters on the screen.
3. Follow the directions for each challenge.

Challenges

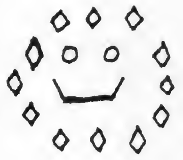
1. Substitute letters for the control graphics characters to decode the following message. The first word is done for you.



The

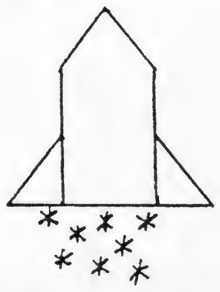
2. Use these keys:
   . T F G N

   To make:

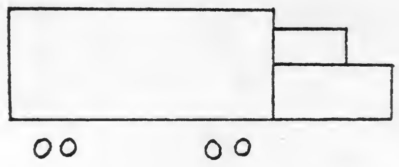


3. Use these keys:
   F G * V B M

   To make:



4. Use these keys:
   Q R E Z C T M A X and (Shift =)
   To make:



CREATE WITH CONTROL GRAPHICS

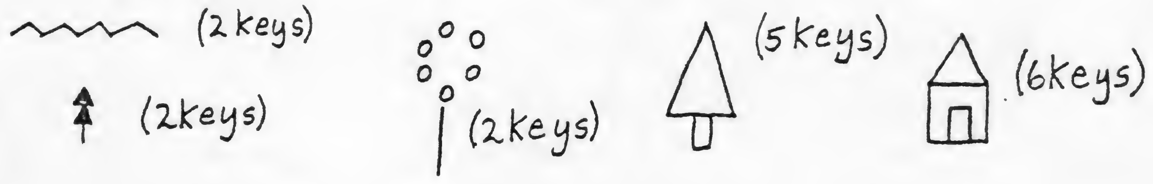
<u>Directions</u>

1. Be sure you are in the memo pad mode before you begin.
2. Lock the keyboard in the graphics character mode by using the CTRL-CAPS/LOWR combination. In order to use the characters on the ",", ".", and ";" keys, you must use the CTRL key. The remainder of the keys will automatically put the graphics characters on the screen.
3. Follow the directions for each challenge.

<u>Challenges</u>

1. Write a coded message to a friend using the control graphics characters. Ask the friend to decode the message. Each character should stand for the letter that shares its key. For example, ✚ = "s", ● = "t", and ▛ = "a", so ✚▛● = sat.

2. See if you can find the keys used to make each of the figures below. When you know how to make each figure, combine them to make a picture. Try adding a coded message as the name of your picture and see if a friend can decode the message.



3. Create your own picture. Use it to write a challenge for another camper or for your teacher.

INSTRUCTIONS FOR PREDICTIONS WORKSHEETS

The items on this worksheet should be a review of things you already know. If you have forgotten any of the information, the computer can be your teacher.

<u>DIRECTIONS</u>

1. In the column called "OUTPUT PREDICTION", write what you think the computer will do when you enter what is written in the "INPUT" column.

2. Type in the input and record the computer's results in the column called "COMPUTER'S OUTPUT".

3. Check your prediction to see if it was correct.

4. If you cannot predict what the results will be, use the computer to help you. Type in the input and see what happens. Then enter the results in the "COMPUTER'S OUTPUT" column.

IMMEDIATE MODE PREDICTIONS

INPUT	OUTPUT PREDICTION	COMPUTER'S OUTPUT
PRINT 16+47		
PRINT 1005-639		
PRINT 14*6		
PRINT 24/12		
PRINT "16+16"		
PRINT 6+4*3		
PRINT 6+(4*3)		
PRINT (6+4)*3		
PRINT 100/20+5		
PRINT 36/(4+5)		
PRINT 36-12+4-8		
PRINT 4+8-36		
PRINT 24/8*12-6		
PRINT 24/8*12-6*10		
PRINT "I'M GREAT!"		

EDITING PROGRAMS

Activity #1

1. Type in the program exactly as it is written. There are 7 stars and 16 X's on line 100. That information may be useful to campers who are having difficulty deciding how to enter the program so that the flag "looks right". The "|" is made with the SHIFT = combination. Make any necessary corrections and the run the program. A transparency and camper copies of the flag and the list of commands in #5 are available. These should not be used without supervision, however.

```
100 PRINT "*******XXXXXXXXXXXXXXXX"
110 PRINT "* * * *               X"
120 PRINT "* * * * XXXXXXXXXXXXXXX"
130 PRINT "* * * *               X"
140 PRINT "XXXXXXXXXXXXXXXXXXXXXXX"
150 PRINT "X                     X"
160 PRINT "XXXXXXXXXXXXXXXXXXXXXXX"
170 PRINT "X                     X"
180 PRINT "XXXXXXXXXXXXXXXXXXXXXXX"
190 PRINT "|"
200 PRINT "|"
210 PRINT "|"
220 PRINT "|"
230 PRINT "|"
240 PRINT "|"
250 PRINT "|"
260 PRINT "|"
270 PRINT "|"
280 END
```

2. List the program and make changes if necessary. If no changes are required, run it and then go on to #3.

3. Add line 90 to the program and run it again.

```
90 PRINT "It's a Grand Old Flag"
```

4. Be sure each camper has a formatted diskette in Drive 1. Save using the command SAVE"D:name".

5. Review the effect of "NEW". Remind campers that "name" is the name they used in #4. Type the following:

```
NEW
LIST
LOAD"D:name"
LIST
RUN
SHIFT-CLEAR
RUN
NEW
LIST
RUN"D:name"
```

EDITING PROGRAMS
Activity #1

```
100 PRINT "******XXXXXXXXXXXXXXXXX"
110 PRINT "* * * *               X"
120 PRINT "* * * * XXXXXXXXXXXXXXX"
130 PRINT "* * * *               X"
140 PRINT "XXXXXXXXXXXXXXXXXXXXXXX"
150 PRINT "X                     X"
160 PRINT "XXXXXXXXXXXXXXXXXXXXXXX"
170 PRINT "X                     X"
180 PRINT "XXXXXXXXXXXXXXXXXXXXXXX"
190 PRINT "|"
200 PRINT "|"
210 PRINT "|"
220 PRINT "|"
230 PRINT "|"
240 PRINT "|"
250 PRINT "|"
260 PRINT "|"
270 PRINT "|"
280 END
```

```
NEW
LIST
LOAD"D:name"
LIST
RUN
SHIFT-CLEAR
RUN
NEW
LIST
RUN"D:name"
```

EDITING PROGRAMS

Activity #2

Campers will use what they practiced in the text editing and control graphics sections. Their challenge is to write a program that will show a picture on the screen. The picture should be constructed from control graphics characters. Text may or may not be included. This is the first exercise that requires planning before using the computer. It is critical that the importance of planning in problem solving and programming be discussed at this point, since structured programming will be emphasized in later activites.

1. Use graph paper to design a picture using the control graphics characters. A general layout or plan is all that is necessary. Some campers may be able to create the picture without preplanning, but it is rare that this is done without some frustration.

2. Use the Basic Utilities Disk. Load the program called "PRINTS". "PRINTS" is a series of print statements with quotation marks. This program should enable campers to make their own pictures or designs much more easily.

3. List the "PRINTS" program and create a picture or design. IT IS EXTREMELY IMPORTANT THAT CAMPERS BE REMINDED TO PRESS RETURN AS THEY FINISH EACH LINE. IF THEY DO NOT, THEIR WORK WILL NOT BE STORED IN MEMORY! Run the program.

4. When campers are happy with their program, instruct them to save it using the SAVE"D:name" command. They should not use "PRINTS" as the name.

5. Practice the following commands:

```
NEW
RUN"D:name"
LIST
NEW
LIST
LOAD"D:name"
```

6. A chart called "RUN, SAVE, LOAD, LIST" is availabe for review purposes. This chart might be most useful to campers who have taken the PILOT class, since the commands are different.

7. Two individual worksheets ("EXPLORE EDITING PROGRAMS" and "CREATE BY EDITING PROGRAMS") are included for campers who need practice, but do not need a formal lesson.

RUN, SAVE, LOAD, LIST

	PILOT	BASIC
Save to disk	SAVE D:NAME	SAVE"D:NAME"
Run program from disk	CAN'T DO	RUN"D:NAME"
Load from disk	LOAD D:NAME	LOAD"D:NAME"
List to printer	SAVE P:	LIST "P:"

EXPLORE EDITING PROGRAMS

DIRECTIONS

1. Type in the program below exactly as it is written.

```
    100 PRINT "            xxx         "
    110 PRINT "            x   x       "
    120 PRINT "            x   x       "
    130 PRINT "            xxx         "
    140 PRINT "              x         "
    150 PRINT "              x  x      "
    160 PRINT "          xxx xx        "
    170 PRINT "              x         "
    180 PRINT "              x         "
    190 PRINT "              x         "
    200 PRINT "             x x        "
    210 PRINT "            x   x       "
    220 PRINT "           x     x      "
    230 PRINT "                        "
    240 PRINT "                        "
    250 PRINT "    X X  XXX      X     "
    260 PRINT "    X X   X       X     "
    270 PRINT "    XXX   X       X     "
    280 PRINT "    X X   X              "
    290 PRINT "    X X  XXX      X     "
```

2. Check for errors, make any necessary changes, and then run the program.

3. List the program. Add line 70 and run it again.
```
     70  REM Stick figure saying, "Hi!".
```

4. List the program again and add the following lines.

```
      90  PRINT "XXXXXXXXXXXXXXXXXXXXXXXXX"
      95  PRINT "A Self Portrait"
      295 PRINT "XXXXXXXXXXXXXXXXXXXXXXXXX"
```

   Run the program. What did lines 90, 95, and 295 do?

5. Save the program on your disk. Use the command SAVE"D:name", where 'name' is the name you give the program.

6. Type NEW and then RUN"D:name".

7. Type NEW and then LOAD"D:name". Now try to run the program.

CREATE BY EDITING PROGRAMS

ECTIONS

1. Use the Basic Utility Disk. Load the program called "PRINTS". The command to load is LOAD"D:PRINTS".

2. "PRINTS" is a series of PRINT statements with quotation marks. This program will enable you to concentrate on the design or picture you decide to make, since all you have to do is type the characters you want to use between the quotation marks. The print statements look like this:

```
100 PRINT "                                        "
200 PRINT "                                        "
```

   and there are lots of them.

3. Use grid paper to plan a design or a picture. You may use the control graphics characters available on the Atari, or the letters on the keyboard. If you use the control characters you won't be able to get a hard copy from the printer.

4. List the "PRINTS" program. Type in the picture or design you made. IT IS EXTREMELY IMPORTANT THAT YOU PRESS RETURN AS YOU FINISH EACH LINE OF THE PROGRAM. IF YOU DO NOT, YOUR WORK WILL NOT BE STORED IN MEMORY!

5. Run your program and make any necessary changes.
6. When you are happy with the program, save it on your disk using the command, SAVE"D:name". "name" is the name you want to give your program.

7. Type NEW and then type LIST. Nothing should be in memory.

8. Run the program from the disk using the command, RUN"D:name".

9. Type NEW again. Now load your program using the command LOAD"D:name". List the program.

10. Finally, send the program to the printer using the command, LIST"P:".

EDITING PROGRAMS
PART II

Introduction

A transparency and camper copies of the code to be typed in (called "EDITING PROGRAMS-PART II") are available. There is one individual activity sheet, "PRACTICE EDITING PROGRAMS", that provides practice for the camper who does not need a formal lesson on this topic.

1. Type in the program exactly as it is written. Check for errors. Make corrections if necessary. Be sure that the campers notice the order in which the lines are entered.

```
NEW
20 PRINT "UP"
10 PRINT "DOWN"
30 PRINT "MIDDLE"
5 PRINT "THIS IS THE BEGINNING"
40 PRINT "THIS IS THE END"
```

2. Run the program noticing the order in which the words were printed on the screen.

3. List the program, again noticing the order in which the lines are printed on the screen.

4. Change lines 10, 20, and 30 (by retyping the whole line) to:

```
10 PRINT "        UP"
20 PRINT "      MIDDLE"
30 PRINT "       DOWN"
```

Run the program.

5. Add the following lines.

```
15 PRINT "LEFT"
25 PRINT "                  RIGHT"
35 PRINT
37 PRINT
```

Run the program.

6. Delete lines 5 and 40. List and then run the program.

EDITING PROGRAMS PART II (CONTINUED)

7. Add a REM to the beginning of the program. Ask campers to compose an appropriate descriptive statement. Discuss the importance of comments, especially in long programs.

8. Run the program. List it again reminding campers that REM statements do not effect the program.

9. Practice listing specified lines to see parts of a program.

10. Change a line number using screen editing and then list the program to see what happens. The safest way to change one line number is shown below.

```
20  PRINT "HELLO"
     Change 20 to 25.
     When you list you get

20  PRINT "HELLO"
25  PRINT "HELLO"

     Type 20 <RETURN>.
     and list the program.
     You should now have

25  PRINT "HELLO"
```

11. Add the END statement at different places in the program to see its effect.

12. Try this line in the program:

```
60 PRINT "This line will help you to remember
how many lines of information you can print
on the screen without adding a new line number"
70 PRINT "You cannot print more than 3 lines.
That is why line 70 was added here."
```

EDITING PROGRAMS
PART II

```
NEW
20 PRINT "UP"
10 PRINT "DOWN"
30 PRINT "MIDDLE"
5 PRINT "THIS IS THE BEGINNING"
40 PRINT "THIS IS THE END"
```

```
10 PRINT "        UP"
20 PRINT "      MIDDLE"
30 PRINT "       DOWN"
```

```
15 PRINT "LEFT"
25 PRINT "                RIGHT"
35 PRINT
37 PRINT
```

```
20  PRINT "HELLO"
     Change 20 to 25.
     When you list you get

20  PRINT "HELLO"
25  PRINT "HELLO"

     Type 20 <RETURN>.
     and list the program.
     You should now have

25  PRINT "HELLO"
```

```
60 PRINT "This line will help you to remember
how many lines of information you can print
on the screen without adding a new line number"
70 PRINT "You cannot print more than 3 lines.
That is why line 70 was added here."
```

PRACTICE EDITING PROGRAMS

1. Type in the program exactly as it is written. Check for errors. Make corrections if necessary. Pay special attention to the order in which the lines are entered.

```
NEW
20 PRINT "THIS IS THE TOP."
10 PRINT "THIS IS THE BOTTOM."
30 PRINT "THIS IS THE MIDDLE."
5 PRINT "FIRST LINE"
40 PRINT "THE END"
```

2. Run the program noticing the order in which the words were printed on the screen.

3. List the program, again noticing the order in which the lines are printed on the screen.

4. Change lines 10, 20 and 30 to:

```
10 PRINT "        TOP"
20 PRINT "      MIDDLE"
30 PRINT "       BOTTOM"
```

5. Add the following lines.

```
15 PRINT "LEFT"
25 PRINT "                RIGHT"
35 PRINT
37 PRINT
```

Run the program.

6. Delete lines 5 and 40. List and then run the program.

7. Add a REM to the beginning of the program. Compose an appropriate statement to describe what the program does.

9. Change a line number using screen editing and then list the program to see what happens. The safest way to change one line number is shown below.

```
100  PRINT "HELLO"
        Change 100 to 105.

        When you list you get
100  PRINT "HELLO"
110  PRINT "HELLO"

        Type 100 <RETURN>.
        and list the program.

        You should now have
110  PRINT "HELLO"
            which is what you want.
```

10. Add the END statement at different places in the program to see its effect.

11. Try this line in the program:

```
60 PRINT "This line will help you to remember
how many lines of information you can print
on the screen without adding a new line number"
70 PRINT "You cannot print more than 3 lines.
That is why line 70 was added here."
```

OPERATORS

An individual worksheet called "PRACTICE WITH OPERATORS" is available for campers who need a review, but do not need a formal lesson. A transparency and individual camper copies of the text for the lesson plan are also available.

1. Type in the following program exactly as it is written.

```
 10 PRINT "GO"
 20 PRINT "STOP"
 30 PRINT
 40 PRINT "GO";
 50 PRINT "STOP"
 60 PRINT
 70 PRINT "GO",
 80 PRINT "STOP"
 90 PRINT
100 PRINT "GO","STOP"
110 PRINT "GO";"STOP"
```

Run the program paying careful attention to the effect of the ";" and the ",".

2. Try adding ","'s and ";"'s to lines. For example:

```
40 PRINT "GO";;;;;
70 PRINT "GO",,,
```

Discuss how these operators might be put to use in a program.

3. Type in the following line exactly as it is written:

```
100 PRINT "FIRST": PRINT "SECOND": PRINT "THIRD"
```

Discuss when it is appropriate to put more than one statement on a line in a program. Emphasize the difficulty of debugging if one has too many statements or unrelated statements together.

4. Try combining ":" with ";" and "," on a line.

5. Give the following summary of the operators:

<u>The Comma ,</u> - Helps organize output into columns. The computer puts words or numbers into columns 10 spaces apart for each comma.

<u>The Semicolon ;</u> - Joins things together.

<u>The Colon :</u> - Used to put more than one instruction in a program line.

OPERATORS

```
 10 PRINT "GO"
 20 PRINT "STOP"
 30 PRINT
 40 PRINT "GO";
 50 PRINT "STOP"
 60 PRINT
 70 PRINT "GO",
 80 PRINT "STOP"
 90 PRINT
100 PRINT "GO","STOP"
110 PRINT "GO";"STOP"
```

```
40 PRINT "GO";;;;;
70 PRINT "GO",,,
```

```
100 PRINT "FIRST": PRINT "SECOND": PRINT "THIRD"
```

The Comma , - Helps organize output into columns. The computer puts words or numbers into columns 10 spaces apart for each comma.

The Semicolon ; - Joins things together.

The Colon : - Used to put more than one instruction in a program line.

PRACTICE WITH OPERATORS

1. Type in the following program exactly as it is written.

```
10 PRINT "HERE"
20 PRINT "THERE"
30 PRINT
40 PRINT "HERE";
50 PRINT "THERE"
60 PRINT
70 PRINT "HERE",
80 PRINT "THERE"
```

Run the program paying careful attention to the effect of the ";" and the ",".

2. Try adding ","'s and ";"'s to lines. For example:

```
40 PRINT "HERE";;;;;
70 PRINT "THERE",,,
```

On the blanks below, write at least one way these operators might be put to use in a program.

____

____

____

____

How can you get a blank between "HERE" and "THERE" in the case where these words are joined?

3. Type in the following line exactly as it is written:

```
100 PRINT "FIRST": PRINT "SECOND": PRINT "THIRD"
```

4. Try combining ":" with ";" and "," on a line.

5. The following is a summary of what each of the operators does:

The Comma , - Helps organize output into columns. The computer puts words or numbers into columns 10 spaces apart for each comma.

The Semicolon ; - Joins things together.

The Colon : - Used to put more than one instruction in a program line.

LPRINT

An individual worksheet called "EXPLORE LPRINT" is available for campers who need practice, but do not need a formal lesson. A transparency and individual camper copies of the programs used in this lesson are also provided.

1. Be sure the printer and (if you have one) the interface are on. Type in the program below.

```
10 REM This program illustrates what LPRINT does
20 PRINT "This line will appear on the screen."
30 LPRINT "THIS LINE WILL NOT APPEAR ON THE SCREEN"
40 LPRINT "LPRINT WORKS JUST LIKE PRINT"
50 LPRINT "EXCEPT THAT IT PRINTS ON THE PRINTER"
```

Run the program. Look carefully at the output to see which lines were output to the screen and which ones were output to the printer.

2. Type in the following program and then run it to further illustrate LPRINT.

```
 10 LPRINT "Said a young, but wise robot
    named Truman,"
 20 PRINT "The instructor's supremacy fades"
 30 LPRINT "'When a man starts fussin'
    and fumin',
 40 PRINT "When robots become teaching aides,"
 50 LPRINT "   And is clumsy and coarse"
 60 PRINT "   And students bring treats"
 70 LPRINT "   I think of the source,"
 80 PRINT "   Of candy and sweets"
 90 LPRINT "And remember he is only human.'"
100 PRINT "To the robot who makes out the grades."
```

3. Combine PRINT and LPRINT in a program as shown in the example below.

```
10 PRINT "SCREEN ";: LPRINT "PRINTER ";
20 PRINT "FIRST": LPRINT "SECOND"
30 PRINT "5 4 3 2 1",
40 LPRINT "5 4 3 2 1",
50 PRINT "BLASTOFF!"
60 LPRINT "BLASTOFF!"
```

Run the program.

LPRINT

```
 10 REM This program illustrates what LPRINT does
 20 PRINT "THIS LINE WILL APPEAR ON THE SCREEN"
 30 LPRINT "THIS LINE WILL NOT APPEAR ON THE SCREEN"
 40 LPRINT "LPRINT WORKS JUST LIKE PRINT"
 50 LPRINT "EXCEPT THAT IT PRINTS ON THE PRINTER"
```

```
 10 LPRINT "Said a young, but wise robot
    named Truman,"
 20 PRINT "The instructor's supremacy fades"
 30 LPRINT "'When a man starts fussin'
    and fumin',
 40 PRINT "When robots become teaching aides"
 50 LPRINT "   And is clumsy and coarse"
 60 PRINT "   And students bring treats"
 70 LPRINT "   I think of the source,"
 80 PRINT "   Of candy and sweets,"
 90 LPRINT "And remember he is only human.'"
100 PRINT "To the robot who makes out the
    grades."
```

```
 10 PRINT "SCREEN": LPRINT "PRINTER"
 20 PRINT "FIRST": LPRINT "SECOND"
 30 PRINT "5 4 3 2 1",
 40 LPRINT "5 4 3 2 1",
 50 PRINT "BLASTOFF!"
 60 LPRINT "BLASTOFF!"
```

EXPLORE LPRINT

1. Be sure the printer and the interface are on. Type in the program below.

```
10 REM This program illustrates what LPRINT does
20 PRINT "THIS LINE WILL OUTPUT TO THE SCREEN"
30 LPRINT "THIS LINE WILL NOT OUTPUT TO THE SCREEN"
40 LPRINT "LPRINT WORKS JUST LIKE PRINT"
50 LPRINT "EXCEPT THAT IT PRINTS ON THE PRINTER"
```

Run the program. Look carefully at the output to see which lines were output to the screen and which ones were output to the printer.

2. Type in the following program and then run it to further illustrate LPRINT.

```
 10 GRAPHICS 2:SETCOLOR 4,13,2
 20 LPRINT "There was a young man from Purdue"
 30 POSITION 5,1:PRINT #6; "##############"
 40 LPRINT "Who dreamed he was eating rock stew."
 50 POSITION 5,2:PRINT #6; "#The  printer#"
 60 LPRINT "   He woke up in the night,"
 70 POSITION 5,3:PRINT #6; "#is  printing#"
 80 LPRINT "   With a terrible fright";
 90 POSITION 5,4:PRINT #6; "##a limerick##"
100 LPRINT "To find it was perfectly true."
110 POSITION 5,5:PRINT #6; "##############"
```

3. Combine PRINT and LPRINT in a program. An example is given below.

```
10 PRINT "SCREEN": LPRINT "PRINTER"
20 PRINT "FIRST": LPRINT "SECOND"
30 PRINT "5 4 3 2 1",
40 LPRINT "5 4 3 2 1",
50 PRINT "BLASTOFF!"
60 LPRINT "BLASTOFF!"
```

Be creative and make up your own program. Save the program on your diskette.

POSITION

1. Review what POSITION does. Include at least the following information:

   - POSITION tells the computer where to start printing on the screen.

   - The format of its use is POSITION 10,3. Remind campers that the 10 tells the number of spaces across, and the 3 tells how many spaces down. A comma must be present between the numerals.

   - In Graphics 0, there are 40 spaces across the screen, numbered 0 to 39. There are 24 spaces down on the screen numbered 0 to 23. Because the numbers start at zero, the first number after POSITION can be from 0 to 39 and the second number from 0 to 23.

2. Practice with POSITION by typing in different numbers in immediate mode. Campers should try to put words or characters on different parts of the screen. The abbreviation POS. could be introduced at this time. Be sure that campers try the following:

```
PRINT "HELLO"
POSITION 0,0:PRINT "HELLO"
```

Call their attention to the fact that in BASIC when one uses a PRINT statement everything is done two spaces from the left edge of the screen.

3. Challenge campers to write a program using POSITION to show the following on the screen:

```
topleft                                   topright


                 My name is __________.
                 I am in the middle.


bottomleft                                bottomright
```

4. Discuss why and how POSITION might be used in a program.

EXPLORE POS.

1. This is a review of what POSITION does. POS. is an abbreviation for POSITION.

- POSITION tells the computer where to start printing on the screen.

- The format of its use is POSITION 10,3. The 10 tells the number of spaces across, and the 3 tells how many spaces down. A comma must be present between the numerals.

- In Graphics 0, there are 40 spaces across the screen, numbered 0 to 39. There are 24 spaces down on the screen numbered 0 to 23. Because the numbers start at 0, the first number after POSITION can be from 0 and 39, and the second number from 0 to 23.

2. Type in the following program and run it.

```
 10 POSITION 0,0: PRINT "   ----->"
 20 POSITION 1,0: PRINT "   ----->"
 30 POSITION 3,0: PRINT "   ----->"
 40 POSITION 5,0: PRINT "   ----->"
 50 POSITION 7,0: PRINT "   ----->"
 60 POSITION 9,0: PRINT "   ----->"
 70 POSITION 11,0: PRINT "   ----->"
 80 POSITION 13,0: PRINT "   ----->"
 90 POSITION 15,0: PRINT "   ----->"
100 POSITION 17,0: PRINT "   ---->"
110 POSITION 19,0: PRINT "   ---->"
```

3. Use POSITION and control graphics to draw a picture on the screen. It would be a good idea to use graph paper to plan this activity before you actually write the program.

MODULE #2 - GRAPHICS AND SOUND

OBJECTIVES

Be familiar enough with graphics modes 0 - 8 to select the proper mode for a task.

Practice using PLOT and DRAWTO.

Know the relationship between COLOR and SETCOLOR and how to use each one.

MATERIALS REQUIRED

BASIC Cartridge
Camper's Personal Diskette
BASIC Utility Disk
Graphics Modes Graph Paper

REFERENCES

<u>Inside Atari Basic</u>, pp. 72-131

<u>Your Atari Computer</u>, pp. 271-290, 325-335

<u>Atari Sound and Graphics</u>, pp. 1-40

<u>Atari 400/800 Basic Reference Manual</u>, pp. 45-58

CONTENT

There are five lessons in the module. The activities are written for the Atari 800. If you use another Atari, be sure to test each lesson. A BASIC cartridge is required for all of the lessons.

<u>Lesson 1 - Graphics 0,1,2</u>

Pages 1-7

New Statements
GRAPHICS (GR.)   PRINT #6;   RUN "D:name"

Prerequisite Statement
POSITION (POS.)

Materials
BASIC Utility Disk: NEAT

Introduces the GRAPHICS statement and the use of modes 0, 1, and 2. Covers the effect of lower case, capitals, and inverse video with PRINT #6. The program, NEAT, on the BASIC Utility Disk demonstrates the difference between GRAPHICS 1 and 2.

Lesson 2 - GRAPHICS 3 THROUGH 8

Pages 8-12

New Statements
PLOT   DRAWTO   COLOR

Prerequisite Statements
GRAPHICS   PRINT #6;   POSITION

Materials
BASIC Utility Disk: CLOWN

Campers experiment with plotting points and drawing lines. The program, CLOWN, on the BASIC Utility Disk gives an excellent comparison of resolution in GRAPHICS 3, 5, and 7. Graphics modes chart gives descriptions and comparisons of each of the modes.

Lesson 3 - SETCOLOR

Pages 13-18

New Statement
SETCOLOR

Prerequisite Statements
GRAPHICS   PLOT   DRAWTO

Materials
BASIC Utility Disk:
CUBE
BOX
COLOR

Campers learn about hue, luminance, and color registers. Three programs on the BASIC Utility Disk give good demonstrations of color: BOX, CUBE, and COLOR.

Lesson 4 - COLOR

Pages 19-25

New Statement
COLOR

Prerequisite Statements

SETCOLOR	PLOT	DRAWTO	GRAPHICS

Activities illustrate how COLOR is used in programs. A chart called "USING COLOR" will help campers understand the relationship between COLOR and SETCOLOR.

Lesson 5 - SOUND

Pages 26-31

New Statements

SOUND	END (Used with SOUND)

Introduces use of SOUND. Campers learn about pitch, distortion, and volume parameters and about sound registers. The focus is on sound effects rather than writing music.

GRAPHICS 0,1,2
(INTRODUCTION)

Before beginning the activities in this lesson, introduce the following items:

1. Graphics 0, 1, and 2 are "text" modes for displaying words on the screen.

2. When the computer is turned on, it is in Graphics 0. Graphics 0 is also entered when SYSTEM RESET is pressed, or when "GRAPHICS 0" is typed.

3. Other graphics modes are entered by typing "GRAPHICS" (or GR.) and the appropriate number. For example, GRAPHICS 2 is entered by typing "GRAPHICS 2" OR "GR. 2".

4. In graphics modes other than 0, there can be a text window. (GTIA modes 9, 10, and 11 don't allow text windows.) Adding 16 to the graphics mode number gets rid of the text window. For example, GRAPHICS 2 + 16 eliminates the window from GRAPHICS 2.

5. POSITION (POS.) can be used in GRAPHICS 0, 1, and 2. Be aware, however, that there are fewer points across and down in GR. 1 and GR. 2.

6. "PRINT #6;" is used in a program to print text on the screen in graphics modes 1 and 2.

7. Control graphics characters are available in GRAPHICS 0. They are used with normal PRINT commands. CTRL-CAPS/LOWR will lock the keyboard into the "graphics character mode." SHIFT-CAPS/LOWR gets the keyboard back to normal. The graphics characters are for screen display. They can not be printed on the printer.

A transparency and student copies of the code used in the teacher lesson plans is available.

GRAPHICS 0,1,2

Activity #1

Remind campers that GR. 1 has a text window at the bottom of the screen, in which letters appear when you type normally. In order to print outside the text window, you must use PRINT #6;. PRINT #6; "Hello" will print the word, "Hello".

The activities in this section deal with GRAPHICS 1 and 2, since these give special capabilities for putting text on the screen. You might want to quickly review the use of control graphics keys in GRAPHICS 0. GRAPHICS 0 is also the mode in which they will be typing in, listing and changing programs.

1. Campers should type in the following lines to see the effect of graphics modes 1 and 2.

```
GR. 1
PRINT "HI!  MY NAME IS ____________"
PRINT #6; "THIS IS GRAPHICS MODE 1."
GR. 2
PRINT #6; "THIS IS"
PRINT #6; "GRAPHICS MODE 2."
PRINT #6; "CAPITAL LETTERS"
PRINT #6; "small letters"
```

Use the inverse video key for the words inside the quotation marks in the next two print statements.

```
PRINT #6; "CAPITAL LETTERS"
PRINT #6; "IN INVERSE VIDEO"
PRINT #6; "small letters"
PRINT #6; "in inverse video"
```

2. Type in the following lines and then experiment with placing different text on the screen in both GRAPHICS 1 and GRAPHICS 2, using a mixture of upper case, lower case, and inverse video. Discuss why "POS. 7,5" appears to be printed in different positions in GR. 1 AND GR. 2.

```
GR. 1: POS. 7,5: PRINT #6; "POS. 7,5"
GR. 2: POS. 7,5: PRINT #6; "POS. 7,5"
```

Activity #1 (continued)

3. Return to Gr. 0, and type in the following program and run it. Campers should type their name in small letters in place of the blank in line 40.

```
10 GRAPHICS 2
20 POS. 4,2: PRINT #6; "************"
30 POS. 4,3: PRINT #6; "*          *"
40 POS. 4,4: PRINT #6; "*__________*"
50 POS. 4,5: PRINT #6; "*          *"
60 POS. 4,6: PRINT #6; "************"
```

Type GR. 0 and list the program. Change line 10 to:

```
10 GRAPHICS 2 + 16
```

add:

```
70 GOTO 70: REM This line keeps the display
on the screen.
```

and run the program again.

4. The program on the BASIC Utility Disk called "NEAT" gives a good demonstration of the use of GRAPHICS 1 and 2. Have the campers run it from the disk by typing:

```
RUN "D:NEAT"
```

Be sure they try both modes 1 and 2 to see the difference.

The code for the program is provided for your information. It would not be appropriate for campers to examine it at this point, since the purpose is demonstration, and not teaching the specifics of the code itself.

Activity #2

Challenge the campers to complete at least one of the activities on the worksheet called "GRAPHICS 0,1,2 - CHALLENGES".

NEAT

```
 10 REM CHARACTER GRAPHICS
 20 PRINT " "
 30 PRINT "GRAPHICS 1 OR 2";
 40 INPUT G
 50 IF G<>1 AND G<>2 THEN 30
 60 GRAPHICS G+16
 70 POSITION 5,3
 80 PRINT #6;"N":REM UPPER CASE N
 90 POSITION 6,4
100 PRINT #6;"e":REM LOWER CASE E
110 POSITION 7,5
120 PRINT #6;"A":REM UPPER CASE INVERSE VIDEO T
130 POSITION 8,6
140 PRINT #6;"t":REM LOWER CASE INVERSE VIDEO T
150 FOR COL=0 TO 3
160 HUE=INT(16*RND(0))
170 FOR LUM=0 TO 14 STEP 2
180 SETCOLOR COL,HUE,LUM
190 FOR PAUSE=1 TO 40:NEXT PAUSE
200 NEXT LUM
210 SETCOLOR COL,HUE,8
220 NEXT COL
230 GOTO 150
240 END
```

GRAPHICS 0,1,2 CHALLENGES

Complete at least one of the challenges below. Show your completed program to the teacher or teaching assistant.

1. Write a program that uses the each of the following in some way. Print some interesting messages in different positions on the screen using a combination of capital letters, small letters, and inverse video.

```
GR. 1 (or 2) + 16
POSITION (POS.)
PRINT #6; "CAPITALS"
PRINT #6; "small"
PRINT #6; "CAPITALS/INVERSE VIDEO"
PRINT #6; "small/inverse video"
GOTO (To keep the display on the screen.)
```

2. Using what you learned in Activity #1, write a program that creates a title page for a book or a computer game. Include a title, author and any other information you think would be appropriate. This does not have to be a real book or game. Use your imagination.

3. Write a program that puts several boxes on the screen in different positions. Change the program so that words are in each of the boxes. Change the color of the words and boxes.

GRAPHICS 0,1,2
(EXERCISES)

```
GR. 1
PRINT "HI!  MY NAME IS ___________"
PRINT #6; "THIS IS GRAPHICS MODE 1."
GR. 2
PRINT #6; "THIS IS"
PRINT #6; "GRAPHICS MODE 2."
PRINT #6; "CAPITAL LETTERS"
PRINT #6; "small letters"
```

Use the inverse video key for the words inside the quotation marks in the next two print statements.

```
PRINT #6; "CAPITAL LETTERS
PRINT #6; "IN INVERSE VIDEO"
PRINT #6; "small letters"
PRINT #6; "in inverse video"
```

```
GR. 1: POS. 7,5: PRINT #6; "POS. 7,5"
GR. 2: POS. 7,5: PRINT #6; "POS. 7,5"
```

```
10 GRAPHICS 2
20 POS. 4,2: PRINT #6; "************"
30 POS. 4,3: PRINT #6; "*          *"
40 POS. 4,4: PRINT #6; "*__________*"
50 POS. 4,5: PRINT #6; "*          *"
60 POS. 4,6: PRINT #6; "************"
```

Type GR. 0 and list the program. Change line 10 to:

```
     10 GRAPHICS 2 + 16
add:
     70 GOTO 70: REM This line keeps the display
     on the screen.
```

GRAPHICS 3 THROUGH 8

Campers should be reminded that in these modes, dots or blocks of color are displayed instead of letters.

A chart called "GRAPHICS MODES CHART" shows the results of using each mode. The chart may be used as an introduction, or to summarize learning after the activities are complete.

It would be appropriate to discuss pixels and resolution at some time during this lesson.

PLOT AND DRAWTO are introduced and/or practiced in this lesson. GOTO and COLOR are used, but campers need not understand how they work until they are formally introduced in subsequent lessons. The REM statement tells what each one does in the program.

A transparency and student copies of the code used in the lesson plans are available. Graphics grids are also available for each mode. The grids are excellent for planning programs and for illustrating the maximum values for X and Y when using PLOT and DRAWTO.

Activities

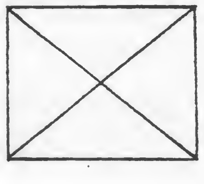
1. Type in the following lines:

```
GR. 3
COLOR 1
PLOT 1,1
PLOT 39,1
PLOT 39,18
PLOT 1,18
PLOT 1,1
```

Notice the position of the squares on the screen, then type:

```
DRAWTO 39,1
DRAWTO 39,18
DRAWTO 1,18
DRAWTO 1,1
```

Challenge campers to make the box look like this:



GRAPHICS 3 THROUGH 8
(CONTINUED)

2. Type in the following program. Ask campers to predict what will be drawn on the screen before they run the program.

```
10 GR. 3 + 16
20 COLOR 1: REM Selects a color for the lines.
30 PLOT 18,1
40 DRAWTO 39,9
50 DRAWTO 18,18
60 DRAWTO 1,9
70 DRAWTO 18,1
80 GOTO 80: REM Keeps the display on the screen.
```

3. Press BREAK and list the program. Explore what happens when the graphics modes are changed and when COLOR is changed. Allow only the numbers 0, 1, 2, or 3 to be used with COLOR.

4. Experiment with PLOT and DRAWTO to become familiar with maximum X and Y values in each mode.

5. A program called "CLOWN", on the BASIC Utility Disk, gives an excellent demonstration of the difference in resolution between graphics modes 3, 5, and 7. It would provide the basis for the discussion of pixels and resolution suggested in the introductory remarks. (The program code is provided for your information.) DO NOT SKIP THIS ACTIVITY.

6. Simulated rainfall. Type in the program. Experiment by changing graphics modes, and by changing the "+" in line 30 to "=", "-", or "*".

```
10 GR. 3+16
20 FOR COUNTER=1 TO 84
30 PRINT #6,"+";
40 NEXT COUNTER
50 GR. 0
60 GOTO 10
```

GRAPHICS 3 THROUGH 8
CAMPER COPY

```
GR. 3
COLOR 1
PLOT 1,1
PLOT 39,1
PLOT 39,18
PLOT 1,18
PLOT 1,1
```

Notice the position of the squares on the screen, then type:

```
DRAWTO 39,1
DRAWTO 39, 18
DRAWTO 1,18
DRAWTO 1,1
```

```
10 GR. 3 + 16
20 COLOR 1: REM Selects a color for the lines.
30 PLOT 18,1
40 DRAWTO 39,9
50 DRAWTO 18,18
60 DRAWTO 1,9
70 DRAWTO 18,1
80 GOTO 80: REM Keeps the display on the screen.
```

Simulated rainfall. Type in the program. Experiment by changing graphics modes, and by changing the "+" in line 30 to "=", "-", or "*".

```
10 GR. 3+16
20 FOR COUNTER=1 TO 84
30 PRINT #6,"+";
40 NEXT COUNTER
50 GR. 0
60 GOTO 10
```

CLOWN

```
 10 DIM CMD$(1)
100 PRINT " "
110 PRINT "MODE 3, 5, OR 7";
120 INPUT MODE
130 GRAPHICS MODE + 16
200 OPEN #1,4,0,"D:CLOWN.DAT":INPUT #1;GR
220 INPUT #1;CMD$:IF CMD$="D" THEN INPUT #1;X,Y:
    GOSUB 500:DRAWTO X,Y:GOTO 220
250 IF CMD$="F" THEN INPUT #1;X,Y,Z:GOSUB 500:
    POSITION X,Y:POKE 765,Z:XIO 18,#6,0,0,"S:"
    PLOT X,Y:GOTO 220
260 IF CMD$="P" THEN INPUT #1,X,Y:GOSUB 500:
    PLOT X,Y:GOTO 220
270 IF CMD$="S" THEN INPUT #1,X,Y,Z:
    SETCOLOR X-1,Y,Z:GOTO 220
280 IF CMD$="C" THEN INPUT #1,X:COLOR X:GOTO 220
300 CLOSE #1
310 GOTO 310
400 END
500 REM SCALING ROUTINE
510 X=X-30
520 IF MODE=7 THEN RETURN
530 IF MODE=5 THEN X=INT(X/2):Y=INT(Y/2)
540 IF MODE=3 THEN X=INT(X/4):Y=INT(Y/4)
550 RETURN
560 END
```

GRAPHICS MODES CHART

MODE	DESCRIPTION	SIZE
GRAPHICS 0	Text mode. Regular type. One color.	40 x 24
GRAPHICS 1	Text mode. Large type. Double width. Five colors.	20 x 20 (split) 20 x 24 (full)
GRAPHICS 2	Text mode. Largest type. Double width. Double height. Five colors.	20 x 10 (split) 20 x 12 (full)
GRAPHICS 3	Large graphics squares. Four colors. Not much memory used. Cannot make detailed drawings.	40 x 20 (split) 40 x 24 (full)
GRAPHICS 4	Smaller graphics points. Two colors, but less memory memory than GR. 5.	80 x 40 (split) 80 x 48 (full)
GRAPHICS 5	Smaller graphics points. Four colors, but uses twice as much memory as GR. 4.	80 x 40 (split) 80 x 48 (full)
GRAPHICS 6	Moderately high resolution Two colors, but uses half as much memory as GR. 7.	160 x 80 (split) 160 x 96 (full)
GRAPHICS 7	Moderately high resolution Four colors, but uses twice as much memory as GR. 6.	160 x 80 (split) 160 x 96 (full)
GRAPHICS 8	High resolution. Two colors. Lots of memory used. Best for detailed drawings.	320 x 160 (split) 320 x 192 (full)

SETCOLOR

This lesson is not meant to be a comprehensive study of SETCOLOR. After completing the activities, campers should know:

1. what SETCOLOR does.
2. the meaning of the words hue and luminance.
3. how to use SETCOLOR to change hue and/or luminance.
4. the appropriate time to use SETCOLOR.
6. that color registers govern the characters, borders, and background colors displayed on the screen.

A transparency and individual copies of the chart called "COLOR CHART" are available for this lesson. (This lesson was written for use with the Atari 800. Some things may be different on the 1200.)

Activity #1 - SETCOLOR

1. Remind campers that the SETCOLOR statement is followed by three numbers. Each number gives the computer information it needs to create the colors you want. The first number is matched with a location in memory and tells what part of the screen display you want to change. The second is the color number and the third controls the luminance. The larger the luminance number, the brighter the color becomes. The value ranges of the parameters are:

Color Register	0-4
Hue	0-15
Luminance	0-14 (Even numbers. Odd numbers are OK, but give the same colors as even numbers.)

2. Experiment with SETCOLOR by typing in:

```
SETCOLOR 2,2,4
```

Change the color and luminance values (the second two numbers) to see what happens. Then change the register number (the first number) to 4 and try different colors and luminances.

3. The BASIC Utility Disk has three programs that illustrate the use of color. Have the campers run each of the following from the disk using RUN "D:______". They should be used in the order listed below.

CUBE - May be used to review the effects of GRAPHICS 3, 5, and 7. Also is an excellent introduction to the range of colors available.

BOX - Gives an excellent demonstration of color and hue. It would be especially good for campers who do not understand the concept of luminance.

COLOR - Manipulates border and screen colors. A good lead in for discussion of how to change the border colors and the screen colors.

A hard copy of the program code is available for your information. It would not be appropriate at this time to discuss the code with the campers.

4. Type in the program listed below. It provides an effective visual definition of the concept of luminance.

```
10 GR. 9
20 SETCOLOR 4,5,0
30 FOR X=0 TO 15
40 COLOR X
50 PLOT X,0
60 DRAWTO X,191
70 NEXT X
80 GOTO 80
```

Have campers change the hue number (5) to see what happens when the color is changed in this program. The color chart would be useful for this activity.

SETCOLOR ACTIVITIES
CAMPER COPY

Experiment with SETCOLOR by typing in:

```
SETCOLOR 2,2,4
```

These are the values you can use:

Color Register	0-4
Hue	0-15
Luminance	0-14

```
10 GR. 9
20 SETCOLOR 4,5,0
30 FOR X=0 TO 15
40 COLOR X
50 PLOT X,0
60 DRAWTO X,191
70 NEXT X
80 GOTO 80
```

CUBE

```
10 REM COLORED CUBE
20 PRINT " ":OPEN #1,4,0,"K:"
30 PRINT "YOU CAN CHANGE THE COLORS OF"
40 PRINT "THE CUBE FACES BY HITTING DIFFERENT"
50 PRINT "KEYS ON THE KEYBOARD."
60 PRINT
70 PRINT "THE CUBE ONLY LOOKS REASONABLE IN"
80 PRINT "GRAPHICS MODES 3, 5, OR 7, BUT"
90 PRINT "YOU CAN TRY OTHER MODES."
100 PRINT "TYPE THE SPACE BAR WHEN YOU WANT"
110 PRINT "TO TRY. A DIFFERENT MODE."
120 PRINT
130 PRINT "GRAPHICS MODE";
140 INPUT G:GRAPHICS G+16
150 FOR I=0 TO 3:SETCOLOR I,0,14:NEXT I:SETCOLOR 4,9,4
160 X=12:Y=9
170 COLOR 1
180 FOR I=0 TO 10
190 PLOT X,Y+I:DRAWTO X+10,Y-I
200 NEXT I
210 COLOR 2
220 FOR I=1 TO 6
230 PLOT X+I,Y-I:DRAWTO X+I+10,Y-I
240 NEXT I
250 COLOR 3
260 FOR I=1 TO 6
270 PLOT X+10+I,Y-I:DRAWTO X+10+I,Y+10-I
280 NEXT I
290 FOR I=0 TO 2
300 GET #1,KEY
310 IF KEY=32 THEN PRINT " ":GOTO 130
320 IF KEY<48 THEN KEY=48
330 SETCOLOR I,1,2*(KEY-48)
340 NEXT I
350 GOTO 290
360 STOP
370 END
```

BOX

```
10 REM WORKSHEET:  COLORED BOX
20 PAUSE=80
30 GRAPHICS 7+16
40 COLOR 1
50 GOSUB 200
60 FOR HUE=0 TO 15
70 FOR LUM=0 TO 14 STEP 2
80 SETCOLOR 0,HUE,LUM
90 GOSUB 300:REM PAUSE
100 NEXT LUM
110 NEXT HUE
120 GOTO 60
200 REM DRAW SQUARE
210 PLOT 90,50
220 DRAWTO 90,30
230 GOSUB 300:REM PAUSE
240 DRAWTO 70,30
250 GOSUB 300:REM PAUSE
260 POSITION 70,50
270 POKE 765,1
280 XIO 18,#6,0,0,"S:"
290 RETURN
300 FOR P=1 TO PAUSE:NEXT P
310 RETURN
320 END
```

COLOR

```
10 REM WORKSHEET: COLOR MANIPULATION
20 REM MANIPULATES BORDER AND DISPLAY SCREEN COLORS.
30 PRINT " ": REM ESC KEY FOLLOWED BY SHIFT CLEAR.
40 REG=2:REM PLAYFIELD 1
50 GOSUB 200
60 SETCOLOR 2,0,0
70 REG=4:REM BACKGROUND
80 GOSUB 200
90 SETCOLOR 4,0,0
100 GOTO 40
200 FOR HUE=0 TO 15
210 FOR LUM=0 TO 14 STEP 2
220 SETCOLOR REG,HUE,LUM
230 FOR PAUSE=1 TO 30:NEXT PAUSE
240 NEXT LUM
250 NEXT HUE
260 RETURN
270 END
```

COLOR CHART

COLOR NUMBER	APPROXIMATE COLOR
0	GRAY
1	GOLD
2	ORANGE
3	RED-ORANGE
4	PINK
5	PURPLE
6	RED-BLUE
7	BLUE
8	BLUE
9	LIGHT BLUE
10	TURQUOISE
11	GREEN-BLUE
12	GREEN
13	YELLOW-GREEN
14	ORANGE-GREEN
15	LIGHT ORANGE

SETCOLOR "DEFAULT" COLORS

REGISTER	COLOR #	LUMINANCE	COLOR
0	2	8	ORANGE
1	12	10	GREEN
2	9	4	DARK BLUE
3	4	6	PINK OR RED
4	0	0	BLACK

COLOR

A paint pot analogy is used to help campers understand COLOR and SETCOLOR. They should learn:

1. what COLOR does.

2. how SETCOLOR and COLOR are related.

3. which color number corresponds to each setcolor number.

4. how to use SETCOLOR and COLOR in a program.

A transparency and student copies of the code used in the lesson are available, as are charts that show how the color registers are used in each mode and how COLOR and SETCOLOR are related.

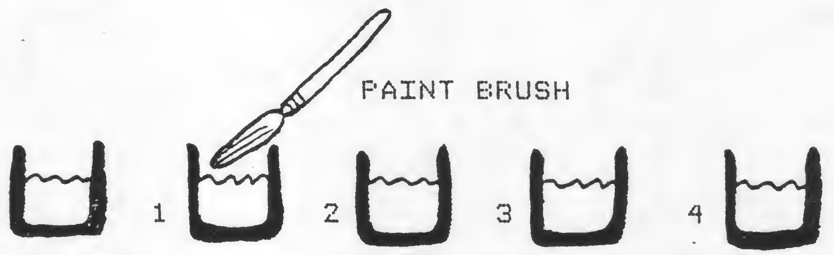
Activity #1

1. Tell campers that COLOR is used to select the color register you want. If you want to change the color of a point or a line to be drawn, you use COLOR and a number which specifies the appropriate color register.

The analogy often used here is that of a paint pot and a paint brush. The color register is manipulated using SETCOLOR. The register is like the paint pot where the paint is stored. COLOR is the brush you use to dip into the pot and draw a point or a line. The page called "USING COLOR" has a drawing that represents the paint pot concept. Use it with campers to continue this discussion.

USING COLOR

PAINT HUES	0=Grey 1=Gold ... 14=Green 15=Orange
PAINT LUMINANCES	0=Dark ... 15=Bright



PAINT POTS	0	1	2	3	4
COLOR CHOICES	First (1)	Second (2)	Third (3)	Fourth (4)	Special (0)

HUE = 0 TO 15
LUM = 0 To 14
POT = 0 To 4

SETCOLOR POT,HUE,LUM	Fills the appropriate pot with the paint of the corresponding HUE and LUM.
COLOR CHOICE	Dips the paint brush into the Corresponding CHOICE. Whatever color is in that POT is the color we will have on our brush. The color in the pot can be changed using SETCOLOR <u>or</u> we can choose a different POT to dip the brush into by using COLOR.

The chart below shows the relationship of the "paint pot" and the "paint brush".

paint pot	paint brush
SETCOLOR 0,_,_	COLOR 1
SETCOLOR 1,_,_	COLOR 2
SETCOLOR 2,_,_	COLOR 3
SETCOLOR 4,_,_	COLOR 0

2. Remind the campers that when you turn the computer on, certain colors are already in the registers. These are called default colors. Type the following to illustrate the default colors.

```
GR. 3
COLOR 1
PLOT 4,2:DRAWTO 36,2
COLOR 2
PLOT 4,4:DRAWTO 36,4
COLOR 3
PLOT 4,6:DRAWTO 36,6
```

3. Go back to GRAPHICS 0 and add line numbers to make a program:

```
10 GR. 3
20 COLOR 1
30 PLOT 2,2:DRAWTO 38,2
40 COLOR 2
50 PLOT 2,4:DRAWTO 38,4
60 COLOR 3
70 PLOT 2,6:DRAWTO 38,6
```

Run the program and then add:

```
35 SETCOLOR 1,10,7
```

and run the program again. To more fully illustrate the relationship of SETCOLOR to COLOR:

a. Change the color and/or hue number (10) in line 35.

b. Add:

```
15 SETCOLOR 0,_,_
55 SETCOLOR 2,_,_
```

filling in the blanks with their choice for color and luminance.

c. Change the number following COLOR in line 40 to 0 and run the program. Challenge campers to change the COLOR numbers to "erase" all of the lines.

COLOR
(CONTINUED)

Activity #2

Review the following information about COLOR and SETCOLOR in the various graphics modes.

GRAPHICS 3, 5, AND 7

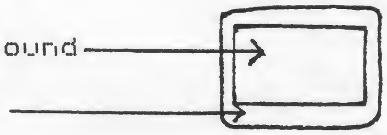
SETCOLOR 4,_,_	Background
SETCOLOR 2,_,_	Text Window
SETCOLOR 0,_,_	COLOR 1
SETCOLOR 1,_,_	COLOR 2
SETCOLOR 2,_,_	COLOR 3

GRAPHICS 4 AND 6

SETCOLOR 4,_,_	Background
SETCOLOR 0,_,_	COLOR 1
SETCOLOR 2,_,_	Text Window

GRAPHICS 8

SETCOLOR 2,_,_	Background
SETCOLOR 4,_,_	Border



The color of the text window is the same color as the background in this mode.

It is of interest to note that the border is always determined by color register 4. The background is determined by register 4 except in GRAPHICS 0 and GRAPHICS 8. In those modes it is register 2 that determines the color.

Activity #3

If the steps in this activity are followed as they are written, the effect of COLOR 0 should be apparent.

1. Type in the following program and run it.

```
10 GR. 5 + 16
20 COLOR 1
30 PLOT 4,4
40 DRAWTO 70,4
50 DRAWTO 70,35
60 DRAWTO 4,35
70 DRAWTO 4,4
90 GOTO 90
```

Add these lines.

```
75 FOR WAIT=1 TO 1000:NEXT WAIT
80 COLOR 0
```

and change line 90 to

```
90 GOTO 30
```

If necessary explain what GOTO does so that campers see that the lines are being erased by drawing them again in the background color.

2. Challenge the campers to write at least one of the following programs.

a. Draw a box whose sides are different colors.

b. Draw a figure in GRAPHICS 8. Make the background color and the border color different from that of the figure and different from each other.

c. Draw a figure that erases itself. Start with GRAPHICS 7 and then change the program to use GRAPHICS 3 and then GRAPHICS 5. Compare the results.

COLOR ACTIVITIES
CAMPER COPY

```
GR. 3
COLOR 1
PLOT 4,2:DRAWTO 36,2
COLOR 2
PLOT 4,4:DRAWTO 36,4
COLOR 3
PLOT 4,6:DRAWTO 36,6
```

```
10 GR. 3
20 COLOR 1
30 PLOT 2,2:DRAWTO 38,2
40 COLOR 2
50 PLOT 2,4:DRAWTO 38,4
60 COLOR 3
70 PLOT 2,6:DRAWTO 38,6
```

Run the program and then add:

```
35 SETCOLOR 1,10,7
```

Run the program again and then add:

```
15 SETCOLOR 0,_,_
55 SETCOLOR 2,_,_
```

filling in the blanks with your choice for color and luminance.

Change the number following COLOR in line 40 to 0 and run the program.

See if you can change the COLOR numbers to "erase" all of the lines on the screen.

COLOR ACTIVITIES
CAMPER COPY

```
10 GR. 5+16
20 COLOR 1
30 PLOT 4,4
40 DRAWTO 70,4
50 DRAWTO 70,35
60 DRAWTO 4,35
70 DRAWTO 4,4
90 GOTO 90
```

Add these lines:

```
75 FOR WAIT=1 TO 1000:NEXT WAIT
80 COLOR 0
```

and change line 90 to:

```
90 GOTO 30
```

CHALLENGE:

Write a program for one of the following:

a. Draw a box whose sides are different colors.

b. Draw a figure in GRAPHICS 8. Make the background color and the border color different from that of the figure and different from each other.

c. Draw a figure that erases itself. Start with GRAPHICS 7 and then change the program to use GRAPHICS 3 and then GRAPHICS 5. Compare the results.

SOUND

The purpose of this lesson is to introduce the use of the SOUND command. When campers finish this lesson they should:

1. know the form of the sound command.

2. know how pitch, distortion, and volume affect a sound that is produced.

3. have added several sound effects subroutines to their subroutine library. These will be provided in the lesson as examples of how sound effects are made.

4. be able to use the SOUND command to create sound effects and musical tones and chords.

Because of the time factor, we will not cover how to write programs that will play music. However, a chart showing the numeric values corresponding to two octaves of musical notes is provided for campers who are musicians and who might like to try some music on their own.

Before you begin, be sure to tell campers that typing "END" will turn off the sound. This is much faster than selecting the proper sound register and setting the values to 0,0,0. They will not use END within their programs. It will be used when experimenting in immediate mode, or after a program is run and the sound is still on.

CAUTION - If you don't like cacophony, this lesson could be hazardous to your health. Before you begin, you might want to establish some ground rules for volume control and signals for your wanting to speak.

<u>Activity #1</u>

1. Experiment with the SOUND command. Try the following:

```
SOUND 0,50,10,6
```

Change the pitch number (50) to see its effect. Then change the distortion (10) and volume (6) numbers to determine what their purpose is. Introduce

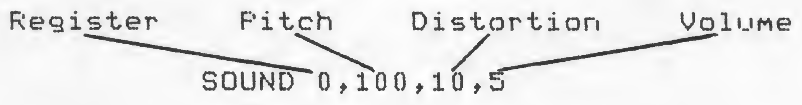
```
SOUND 0,0,0,0
```

to turn off the sound. Then tell campers that typing "END" will also turn off the sound.

2. Now try this.

```
SOUND 0,100,10,5
SOUND 1,150,10,5
SOUND 2,200,10,5
SOUND 3,250,10,5
```

Talk about the four sound registers, numbered 0 - 3, that may be used to make sounds. It may be compared to using four part harmony when singing songs. Each of the registers would be a voice. Tell what the four numbers represent.



The values that can be used in each position are:

Register	0 - 3
Pitch	0 - 255
Distortion	0 - 14 (Even numbers)
Volume	0 - 15

Talk about pitch and distortion, keeping in mind that there will be other examples in the lesson to help campers understand what these terms mean.

<u>Activity # 2</u>

1. Try this program and then discuss what it does. If the group does not understand variables, do not do an in depth lesson on that concept. The important thing here is that they know that the value in the position that selects the pitch is changing.

```
10 FOR PITCH=0 TO 255
20 SOUND 0,PITCH,10,10
30 NEXT PITCH
40 END
```

Add:

```
40 FOR PITCH=255 TO 0 STEP -1
50 SOUND 0,PITCH,10,10
60 NEXT PITCH
70 END
```

and run the program.

SOUND
(CONTINUED)

2. Change the program in #1 so that the distortion values are different and run it. Discuss the effect of distortion and point out that a value of 8 or 10 produces pure tones that we use in writing music.

3. Type in the following and run it for fun.

```
100 FOR COUNT=1 TO 3
200 FOR PITCH=1 TO 255
300 POKE 710,P:SOUND 0,P,10,5
400 FOR WAIT=1 TO 5:NEXT WAIT
500 NEXT PITCH
600 NEXT COUNT
```

Activity #3

On the BASIC Utility Disk there is a program called "SOUNDS" that might help some students who do not understand the meaning of pitch and distortion. Run it from the disk using the command, RUN "D:SOUNDS", and discuss how the sounds changed as the input to the program changed.

Activity #4

If campers want to try to write a short tune, they could do so as an individual activity. This can be a very time consuming process, but may be of interest to some. A delay loop must be used after each "note" played in order to hear the note. An example follows:

```
10 SOUND 0,81,10,8
20 FOR DELAY=1 TO 300:NEXT D
30 SOUND 0,64,10,8
40 FOR DELAY=1 TO 100:NEXT D
50 SOUND 0,53,10,3
60 FOR DELAY=1 TO 500:NEXT D
70 SOUND 0,64,10,8
```

A chart called "MUSICAL NOTES" is available for use in this project. A more comprehensive chart is on page 391 of Your Atari Computer.

NOTE: Delay loops are very tricky in BASIC because a FOR..NEXT loop near the top of a program executes much faster than one near the bottom. Timing can be made more consistent by using the same FOR..NEXT loop, which is in a subroutine for the sole purpose of creating a delay. In this activity, the delay loops were purposely inserted after each line with a SOUND command to visually reinforce the fact that a delay must occur after each "note" in order to hear it.

MUSICAL NOTES

	PITCH	VALUE
	C	60
	B	64
	A or B	68
	A	72
	G or A	76
HIGH	G	81
NOTES	F or G	85
	F	91
	E	96
	D or E	102
	D	108
	C or D	114
MIDDLE	C	121
	B	128
	A or B	136
	A	144
	G or A	153
	G	162
	F or G	173
LOW	F	182
NOTES	E	193
	D or E	204
	D	217
	C or D	230
	C	243

EXPERIMENT WITH SOUND
(CAMPER COPY)

1. Experiment with the SOUND command. Try the following:

```
SOUND 0,50,10,6
```

Change the pitch number (50) to see its effect. Then change the distortion (10) and volume (6) numbers to determine what their purpose is. Use:

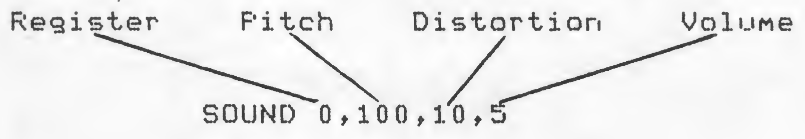
```
SOUND 0,0,0,0
```

to turn off the sound.

2. Now try this.

```
SOUND 0,100,10,5
SOUND 1,150,10,5
SOUND 2,200,10,5
SOUND 3,250,10,5
```

There are four sound registers, numbered 0 - 3, that may be used to make sounds. It may be compared to using four part harmony when singing songs. Each of the registers would be a voice. This is what the four numbers represent.



The values that can be used in each position are:

Register	0 - 3
Pitch	0 - 255
Distortion	0 - 14 (Even numbers)
Volume	0 - 15

SOUND ACTIVITIES
(CAMPER COPY)

```
10 FOR PITCH=0 TO 255
20 SOUND 0,PITCH,10,10
30 NEXT PITCH
40 END
```

Add:

```
40 FOR PITCH=255 TO 0 STEP -1
50 SOUND 0,PITCH,10,10
60 NEXT PITCH
70 END
```

and run the program.

```
100 FOR COUNT=1 TO 3
200 FOR PITCH=1 TO 255
300 POKE 710,P:SOUND 0,P,10,5
400 FOR WAIT=1 TO 5:NEXT WAIT
500 NEXT PITCH
600 NEXT COUNT
```

```
10 SOUND 0,81,10,8
20 FOR DELAY=1 TO 300:NEXT D
30 SOUND 0,64,10,8
40 FOR DELAY=1 TO 100:NEXT D
50 SOUND 0,53,10,3
60 FOR DELAY=1 TO 500:NEXT D
70 SOUND 0,64,10,8
```

MODULE #3 - VARIABLES

OBJECTIVES

Know what a variable is.

Know how to use numeric and string variables.

Be able to write a program using INPUT.

Know how to generate random numbers using RND and INT.

MATERIALS REQUIRED

BASIC Cartridge
Camper's Personal Diskette

REFERENCES

Inside Atari Basic, pp. 40-44

Your Atari Computer, pp. 64-65, 83-95, 108-110

CONTENT

There are four lessons in this module. Campers will be saving more of the programs that they use. No screen formatting has been done in the programs. That is left for the campers to do as an exercise. GOTO is included even though it has not been formally introduced.

Lesson 1 - Variables

Pages 1-3

New Statements
DIM NAME$="_______"

Materials
BASIC Cartridge

The concept of a variable is the focus of the lesson. Campers should know that a variable is the name for a place into which values may be stored. Both numeric and string variables are introduced.

Lesson 2 - LET

Pages 4-7

New Statements
LET   ? (for PRINT)

Prerequisite Statement
DIM

Materials
BASIC Cartridge
Camper's Personal Diskette

Introduces the LET statement. Reviews variables and covers the rules for legal variable names.

Lesson 3 - INPUT

Pages 8-14

New Statements
INPUT   ? #6;

Prerequisite Statements
COLOR   DIM   ?   GR.   SETCOLOR

Materials
BASIC Cartridge
Camper's Personal Diskette

Campers are given a varitey of programs to show how INPUT allows the person using the program to put in their own values. Uses numeric and string variables. Ends with several programming challenges.

Lesson 4 - FUNCTIONS RND & INT

Pages 15-16

New Statements
RND   INT

Materials
BASIC Cartridge

Gives a brief introduction to the RND and INT functions.

VARIABLES

Even if students have had some experience using variables, it would be beneficial to do the activities in this lesson. It is possible to use variables in programs and yet not understand the underlying concept of a variable. If campers do not have a clear understanding of variables, it is likely that they will be hopelessly confused when they are introduced to arrays and other data structures.

The concept of a variable is the focus of this lesson. Students should know that a variable is a name for a place into which values may be stored.

Activity #1

1. This activity should help campers remember that a variable is the name of a place where a value is stored. Have half of the group be variable demonstrators. We will only consider numbers at this point.

2. Use the campers' names as the variable names. Have other campers give a numeric value (written on a piece of paper) to each camper who is a variable demonstrator. As each value is delivered say, "The variable named _______ now has the value of ___." Explain that variables are assigned (or given) values. They are not the same as (= to) the value they are given. You might even say, for example, that Debbie is not equal to 4. She has been given the value of 4.

3. Next, deliver new values to the variables. Explain that the old values are to be thrown away before the new values are assigned. Have the demonstrators actually throw away the paper with the old value.

4. Now use the computer in immediate mode to demonstrate the same concept. Type in the following:

```
PRINT NUM
NUM=10
PRINT NUM
NUM=1000
PRINT NUM
NUM=NUM+NUM
PRINT NUM
```

Point out that the value 0 is stored in a numeric variable when the machine is turned on.

VARIABLES
(CONTINUED)

Activity #2

1. In this activity campers will use string variables as well as numeric variables. A drawing called "VARIABLES - ACTIVITY #2" is provided.

2. Use the drawing to show the difference in the forms of string and numeric variables. Be sure to point out that variable names may not contain a space. Thus, HOWMANY is one word. The drawing is designed to illustrate the fact that numeric variables only have one box. String variables must have a box for each character and the number of boxes necessary must be stated in a program before the string variable is used.

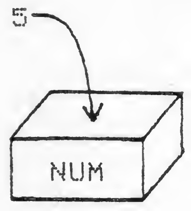
3. Fill in the boxes labeled NUM and HOWMANY with larger numbers. Have campers suggest another name to put in the box labeled NAME$ and an answer to a question that is nine letters or less to put in the box labeled ANSWER$. Explain the DIM to the right of each string variable box. When you change the values in each of the boxes, reinforce the fact that when a new value is assigned to a variable, the old value is "thrown out".

4. Use the computer in immediate mode to demonstrate how to use string variables. Instruct campers that when an assignment is made to a string variable, quotation marks must be used. Type in the following:

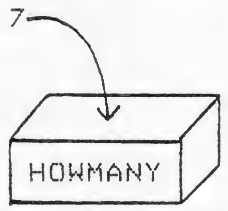
```
DIM NAME$(6)
NAME$="GEORGE"
PRINT NAME$

NAME$="CAROL"
PRINT NAME$
```

Try a name that has more than 6 letters to see the error in dimension.

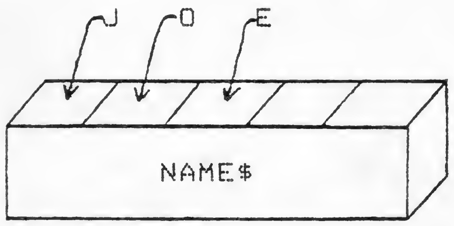
VARIABLES — ACTIVTIY #2



NUM=5

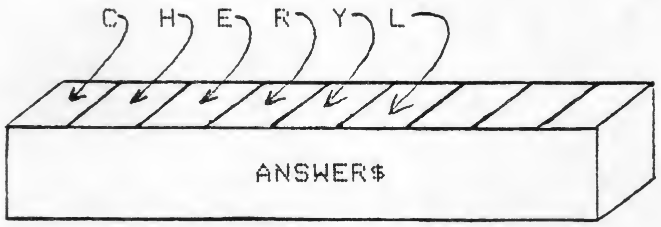


HOWMANY=7



DIM NAME$(5)

NAME$="JOE"



DIM ANSWER$(9)

ANSWER$="CHERYL"

```
DIM NAME$(6)
NAME$="GEORGE"
PRINT NAME$

NAME$="CAROLE"
PRINT NAME$
```

LET

Review variables before beginning the activities in this lesson. Be sure that campers know the following:

1. Numeric variables may be used only with numbers.

2. String variables can store any character. They must be given a dimension and the value assigned to the variable must be enclosed in quotation marks.

3. Variable names must start with a letter. Reserved words such as LIST, RUN, PRINT, LOAD, etc. may not be used as variables.

Introduce the LET statement. It is important that LET be used at first, so that campers are reminded that LET A=5 means "Let A be given the value of 5", not "A equals 5". Do not skip the discussion of each program. That is when learning will take place for most students.

A transparency and individual student copies of the code used in the activities are available.

Activity #1

Type in the following program and run it. Instruct campers to use the BREAK key to stop the program. Discuss what happens to the variable COUNT in each line.

```
 5 REM A counting program.
10 LET COUNT=0
20 LET COUNT=COUNT+1
30 PRINT COUNT
40 REM Line 50 says go back to line 20.
50 GOTO 20
```

If necessary, review the effect of GOTO in line 50.

<u>Activity #2</u>

This program provides practice using numeric variables. Type it in and run it. Then discuss what happens at each line. The "?" is used for the first time to mean PRINT. Discuss this abbreviated PRINT statement with the campers before they begin.

(Type NEW to clear memory.)

```
 5 REM Practice using numeric variables.
10 LET NUM1=5
20 LET NUM2=7
30 LET NUM3=NUM1+NUM2
40 LET NUM4=NUM1*NUM2
50 ? "NUM1 = ";NUM1
60 ? "NUM2 = ";NUM2
70 ? NUM1;"+";NUM2;"=";NUM3
80 ? NUM1;"*";NUM2;"=";NUM4
```

Ask campers to change the values in NUM1 and NUM2 and run the program again. To further illustrate what happens with variables, change NUM3 in line 70 to NUM4 and run the program again.

<u>Activity #3</u>

Practice using string variables. Campers should use their own name in line 30. Be sure to leave a space before the word "is" in line 40.

(Type NEW to clear memory.)

```
 5 REM Practice using string variables.
10 DIM NAME$(15)
20 LET NAME$="____________"
30 ? NAME$;
40 ? " is learning about variables."
```

Change the name used in line 20 and run the program again.

Activity #4

This activity illustrates the fact that numerals can be printed as strings, but they must have quotation marks around them to tell the computer that they are numerals and not numbers. Any name may be used in line 20.

(Type NEW to clear memory.)

```
 5 REM Using numbers in strings.
10 DIM NAME$(15)
20 LET NAME$="__________"
30 ? "HI! MY NAME IS "; NAME$;"."
40 ? "SEE HOW FAST I CAN COUNT TO 10!"
50 DIM NUMS$(25)
60 LET NUMS$="1 2 3 4 5 6 7 8 9 10"
70 ? NUMS$
```

ACTIVITY #5

This activity shows that assignment may be made without using LET. It may be necessary to discuss what an average is before or after the program is run.

(Type NEW to clear memory.)

```
 5 REM Assignment without LET; Calculating averages
10 N1=7
20 N2=8
30 N3=10
40 N4=105
50 N5=1000
60 N6=(N1+N2+N3+N4+N5)/5
70 ? "The average of "; N1;", ";N2;", ";N3;
   ", ";N4;", and ";N5; " is ";N6;"."
```

LET
(PROGRAMS FOR PRACTICE)

```
 5 REM A counting program.
10 LET COUNT=0
20 LET COUNT=COUNT+1
30 PRINT COUNT
40 REM Line 50 says go back to line 20.
50 GOTO 20
```

```
 5 REM Practice using numeric variables.
10 LET NUM1=5
20 LET NUM2=7
30 LET NUM3=NUM1+NUM2
40 LET NUM4=NUM1*NUM2
50 ? "NUM1 = "; NUM1
60 ? "NUM2 = "; NUM2
70 ? NUM1;"+";NUM2;"=";NUM3
80 ? NUM1;"*";NUM2;"=";NUM4
```

```
 5 REM Practice using string variables.
10 DIM NAME$(15)
20 LET NAME$="____________"
30 ? NAME$;
40 ? " is learning about variables."
```

```
 5 REM Using numbers in strings.
10 DIM NAME$(15)
20 LET NAME$="__________"
30 ? "HI! MY NAME IS ";NAME$;"."
40 ? "SEE HOW FAST I CAN COUNT TO 10!"
50 DIM NUMS$(25)
60 LET NUMS$="1 2 3 4 5 6 7 8 9 10"
70 ? NUMS$
```

```
 5 REM Assignment without LET; Averaging.
10 N1=7
20 N2=8
30 N3=10
40 N4=105
50 N5=1000
60 N6=(N1+N2+N3+N4+N5)/5
70 ? "The average of "; N1;", ";N2;", ";N3;
   ", ";N4;", and ";N5; " is ";N6;"."
```

INPUT

Campers will be given a variety of programs to show how INPUT allows the person using the program to put in their own values.

It is necessary that numeric and string variables be understood before doing this lesson.

A transparency and individual student copies of the code used in the activities is available.

Activity #1

Type in and run the following program.

```
 5 REM Asks for a name. Uses string input.
10 DIM NAME$(20)
20 ? "What is your first name";
30 INPUT NAME$
40 ? "HI, ";NAME$;".  I'm happy to make
   your acquaintance."
```

Try the program without line 20 to show the importance of using a question or statement to prompt the user for the proper information.

Activity #2

Refer to the program used for Activity #5 in the LET lesson. It averages five numbers. In this activity, campers will type in a program that allows them to select the five numbers to be averaged.

```
 5 REM Averaging program with INPUT.
10 ? "This program will calculate the average"
20 ? "of any five numbers you choose.  Type the"
30 ? "numbers in with commas between them--"
40 ? "like this 10,3,40,70,90."
50 INPUT NUM1,NUM2,NUM3,NUM4,NUM5
60 AVE=(NUM1+NUM2+NUM3+NUM4+NUM5)/5
70 ? "The average of your numbers is ";AVE;"."
```

Ask campers to add lines that will print the numbers that were entered as well as the average of the numbers.

Activity #3

This program accepts a numeric INPUT and calculates a person's age in the year 2001.

```
  5 REM Calculates age in the year 2001
 10 ? "Have you ever wondered how old you"
 20 ? "will be in the year 2001?"
 30 ? "This program will do the necessary"
 40 ? "calculations for you very quickly."
 50 ? "Enter your age:";
 60 INPUT AGE
 70 ? "What year is it?  (Be sure to use all"
 80 ? "four numbers, eg. 1983)";
 90 INPUT YEAR
100 ANSWER=2001-YEAR+AGE
110 ? "In 2001, you will be ";ANSWER;" years old."
```

Challenge campers to change the program, so that a user may input another year besides 2001.

Activity #4

There are two programs in this activity that use INPUT. In one, a graphics mode (0, 1, or 2) may be selected and a word that appears on the screen may be changed. The second program involves changing values in the SETCOLOR command. If campers are not familiar with Atari graphics, the programs will need some explanation.

The first program changes the background color in GRAPHICS 3.

```
  5 REM  This program allows the user to
 10 REM  change the hue and luminance values
 20 REM  in a SETCOLOR command.
 30 ? "In this program you will be able to"
 40 ? "change the color of the background on"
 50 ? "the screen."
 60 ? "Choose a number (0-15) for the hue";
 70 INPUT HUE
 80 ? "Choose a number (0-14) for the luminance";
 90 INPUT LUM
100 GR. 3
110 SETCOLOR 4,HUE,LUM
```

The next program allows the graphics mode and text displayed on the screen to be changed.

```
  5 REM  This program allows the user to
 10 REM  change the graphics mode to
 20 REM  display different sizes of text on
 30 REM  on the screen.  It also allows the
 40 REM  user to change the text on the screen.
 50 ? "In this program, you will be able to"
 60 ? "enter a number from 0 to 2 for the"
 70 ? "graphics mode and a word that you"
 80 ? "want to display."
 90 ? "What graphics mode would you like";
100 INPUT MODENUM
120 ? "You may display any word up to 15 letters."
130 ? "What word would you like to display";
140 DIM WORD$(15)
150 INPUT WORD$
160 GR. MODENUM
170 COLOR 2
180 POS. 5,6
190 ? #6;WORD$
```

Activity #5

This program uses INPUT with string variables to create a sentence generator. Discuss the fact that all the string variables are given dimensions at the beginning of the program and with one DIM statement.

```
  5 REM This program uses string variables to
 10 REM create a sentence generator.
 20 DIM NOUN$(15),VERB$(15),ADJ$(15),NOUN2$(15)
 30 ? "Please enter a plural noun:";:INPUT NOUN$
 40 ? "Now enter a verb that goes with a"
 50 ? "plural noun:";:INPUT VERB$
 60 ? "Next enter an adjective.  Remember"
 70 ? "that an adjective is a word"
 80 ? "that describes a noun: ";:INPUT ADJ$
 90 ? "Finally, enter another plural noun: ";
100 INPUT NOUN2$
110 ? "Did you know that ";
120 ? NOUN$;" ";VERB$;" ";
130 ? ADJ$;" ";NOUN2$;"?"
```

Run the program and see if the sentence generated makes sense. If it does not, have campers try to input words that will make sense. Then ask if they can rewrite the program to be sure that proper sentences are generated.

INPUT
CAMPER COPY

```
 5 REM Asks for a name. Uses string input.
10 DIM NAME$(20)
20 ? "What is your first name";
30 INPUT NAME$
40 ? "HI, ";NAME$;".  I'm happy to make
   your acquaintance."
```

```
 5 REM Averaging program with input.
10 ? "This program will calculate the average"
20 ? "of any five numbers you choose.  Type the"
30 ? "numbers in with commas between them--"
40 ? "like this 10,3,40,70,90."
50 INPUT NUM1,NUM2,NUM3,NUM4,NUM5
60 AVE=(NUM1+NUM2+NUM3+NUM4+NUM5)/5
70 ? "The average of your numbers is ";AVE;"."
```

```
  5 REM Calculates age in the year 2001
 10 ? "Have you ever wondered how old you"
 20 ? "will be in the year 2001?"
 30 ? "This program will do the necessary"
 40 ? "calculations for you very quickly."
 50 ? "Enter your age:";
 60 INPUT AGE
 70 ? "What year is it?  (Be sure to use all"
 80 ? "four numbers, eg. 1983)";
 90 INPUT YEAR
100 ANSWER=2001-YEAR+AGE
110 ? "In 2001, you will be ";ANSWER;" years old."
```

INPUT
(CAMPER COPY CONTINUED)

```
  5 REM  This program allows the user to
 10 REM  change the hue and luminance values
 20 REM  in a SETCOLOR command.
 30 ? "In this program you will be able to"
 40 ? "change the color of the background on"
 50 ? "the screen."
 60 ? "Choose a number (0-15) for the hue";
 70 INPUT HUE
 80 ? "Choose a number (0-14) for the luminance";
 90 INPUT LUM
100 GR. 3
110 SETCOLOR 4,HUE,LUM
```

```
  5 REM  This program allows the user to
 10 REM  change the graphics mode to
 20 REM  display different sizes of text on
 30 REM  on the screen.  It also allows the
 40 REM  user to change the text on the screen.
 50 ? "In this program, you will be able to"
 60 ? "enter a number from 0 to 2 for the"
 70 ? "graphics mode and a word that you"
 80 ? "want to display."
 90 ? "What graphics mode would you like";
100 INPUT MODENUM
120 ? "You may display any word up to 15 letters."
130 ? "What word would you like to display";
140 DIM WORD$(15)
150 INPUT WORD$
160 GR. MODENUM
170 COLOR 2
180 POS. 5,6
190 ? #6;WORD$
```

INPUT
(CAMPER COPY CONTINUED)

```
  5 REM This program uses string variables to
 10 REM create a sentence generator.
 20 DIM NOUN$(15),VERB$(15),
 ADJ$(15),NOUN2$(15)
 30 ? "Please enter a plural noun:";:INPUT NOUN$
 40 ? "Now enter a verb that goes with a"
 50 ? "plural noun:";:INPUT VERB$
 60 ? "Next enter an adjective.  Remember"
 70 ? "that an adjective is a word"
 80 ? "that describes a noun:";:INPUT ADJ$
 90 ? "Finally, enter another plural noun:";
100 INPUT NOUN2$
110 ? "Did you know that ";
120 ? NOUN$;" ";VERB$;" ";
130 ? ADJ$;" ";NOUN2$;"?"
```

PROGRAMMING CHALLENGES USING VARIABLES

Choose one the the following tasks and write a program that completes it.

1. Ask the user to enter five words. Print the nursery rhyme below using the words as input. Specify the kinds of words you want, so that when you use them to fill in the blanks, the nursery rhyme will make sense.

> Little Miss ________________
> Sat on a _________________
> Eating her ___________ and ______________.
> Along came a _________________
> Who sat down beside her
> And frightened Miss ______________ away.

2. Compute a player's batting average when given the number of times at bat, number of hits and number of walks.

3. Save youself time by writing one form letter that you can send to several people by just changing the greeting. If you want to get fancy, you could also change some of the words in the letter. The following is a short example:

> Dear ___________,
>
> How are you, ___________? I am having a great time at Atari Summer Camp.
>
> __________,
> Sandy

This could also be used to send a "personalized" message to everyone in camp.

4. Ask the user to enter the appropriate dimensions and then compute the area of a geometric figure such as a square, rectangle, triangle, circle, or parallelogram.

5. Use INPUT, LET, at least 3 string variables, and 2 numeric variables to write a program on a topic of your choosing.

FUNCTIONS RND & INT

This lesson is a brief introduction to the RND and INT functions. It is presented in this module because many of the programs in subsequent modules include these functions. Upon completion of the lesson, campers should be able to use the functions to generate random numbers.

Activity #1

Use immediate mode to experiment with both RND and INT. Have campers try the following entries. Compare the numbers each of them gets to show the randomness.

```
PRINT RND(0)*4
PRINT RND(0)*10
PRINT RND(0)*50
```

Now do the same with INT and discuss the results.

```
PRINT INT(1.9)
PRINT INT(30.111)
PRINT INT(0.65)
PRINT INT(100000.9)
```

Activity #2

1. Use RND in a program to show the purpose of the parts of the function.

```
10 NUM=RND(0)*5
20 PRINT NUM
30 GOTO 10
```

Run the program and examine the results.

2. Change the 5 to other numbers. Note the decimal numbers and the fact that the output ranges from 0 to one less than the number multiplied times RND(0).

3. Change line 10 to:

```
10 NUM=RND(0)*5+1
```

Run the program again and note the difference in the output. Change line 10 again to:

```
10 NUM=INT(RND(0)*5+1)
```

Run the program. Talk about the possible uses of RND and INT in writing programs.

FUNCTIONS RND & INT
(CAMPER COPY)

```
PRINT RND(0)*4
PRINT RND(0)*10
PRINT RND(0)*50
```

```
PRINT INT(1.9)
PRINT INT(30.111)
PRINT INT(0.65)
PRINT INT(100000.9)
```

```
10 NUM=RND(0)*5
20 PRINT NUM
30 GOTO 10
```

```
10 NUM=RND(0)*5+1
```

Run the program again and note the difference in the output.
Change line 10 again to:

```
10 NUM=INT(RND(0)*5+1)
```

MODULE #4 - LOOPS

OBJECTIVES

Be able to write a program using FOR..NEXT.

Given a nested loop, be able to predict the output.

MATERIALS REQUIRED

BASIC Cartridge
Camper's Personal Diskette

REFERENCES

Inside Atari Basic - pp. 45-52

Your Atari Computer - pp. 80-86

Atari 400/800 Basic Reference Manual - pp. 15-17

CONTENT

Campers must know how to use variables to do the activities in the module's four lessons. A BASIC cartridge is required for all lessons.

Lesson 1 - Loops - FOR..NEXT

Pages 1-3

New Statement
FOR..NEXT

Covers counting loops, delay loops, and using variables as upper and lower limits in loops.

Lesson 2 - Loops - FOR..NEXT..STEP

Pages 4-6

New Statement
FOR..NEXT..STEP

Introduces ESC CTRL-CLEAR to clear screen. Uses FOR..NEXT to assign values to the SOUND statement.

Lesson 3 - Nested Loops

Pages 7-11

Shows how to write a program to simulate a TAB function. Introduced diagramming loops to see if they are nested properly.

Lesson 4 - GOTO

Pages 12-13

New Statement
GOTO

"The Allowance Con" illustrates the use of GOTO. In the program they ask for an allowance of one cent the first week. They then ask that the amount be doubled each week thereafter with surprising results.

LOOPS
FOR...NEXT

FOR...NEXT loops have been part of the code in several programs used as examples in previous modules. Campers have seen them, but may not be aware of how they work or when it is appropriate to use them.

In order to do the lesson, campers should know how to use variables. Transparencies and individual copies of the code used in the lessons are available.

Activity #1

Type in the following program and run it.

```
10 REM FOR...NEXT Introduction
20 FOR COUNT=1 TO 10
30 PRINT COUNT
40 NEXT COUNT
```

Discuss how the variable COUNT in line 30 gets its value, and how the loop cycles through. Then change line 20 to:

```
20 FOR COUNT=1 TO 20
```

and ask campers to tell what the program will do before they run it. Change line 20 to:

```
20 FOR COUNT=5 TO 15
```

and run the program again. Have campers experiment with changing the numbers in line 20 to see what will work in the loop.

Activity #2

Type in the following program and run it.

```
10 REM Printing lots of letters.
20 FOR NUMTIMES=1 TO 400
30 PRINT "Z-";
40 NEXT NUMTIMES
50 PRINT "WOOPS! TIME TO WAKE UP."
```

Challenge campers to print their name on the screen in different patterns using FOR...NEXT loops and commas.

Activity #3

The program in this activity will use variables for the lower and upper limits of the FOR...NEXT loop. Enter and run the following:

```
10 REM Using variables and limits
20 START=1
30 FOR NUM=START TO 10
40 PRINT NUM
50 NEXT NUM
```

Discuss what START does. Change the value assigned to START and run the program. Challenge campers to change the program, so that it asks for input from the user and then uses the input to determine the lower and upper limits in the loop. Call the upper limit "FINISH".

Activity #4

The following program illustrates the danger of incrementing the same variable inside a loop that is used in the loop. Enter and run it. Discuss how COUNTER got its value at each step in the loop.

```
10 REM Dangerous and improper
15 REM use of a variable
20 FOR COUNTER=1 TO 25
30 COUNTER=COUNTER+1
40 PRINT COUNTER
50 NEXT COUNTER
```

Activity #5

Type in the following program and run it.

```
10 REM Illustration of a Delay Loop
20 PRINT "Please wait a moment."
30 FOR WAIT=1 TO 3000:NEXT WAIT
40 PRINT "Thank you for waiting."
```

Discuss the fact that a FOR...NEXT loop can tell the computer to do nothing for a certain number of times. Since the computer is so fast, relatively large numbers must be used in the loop. Have campers experiment with the upper limit to see the effect of different values.

NOTE: Delay loops are very tricky in BASIC. A FOR..NEXT loop at the top of a program executes much faster than one near the bottom. Timing can be made more consistent by putting the loop in a subroutine.

LOOPS
FOR...NEXT
(CAMPER COPY)

```
10 REM FOR...NEXT Introduction
20 FOR COUNT=1 TO 10
30 PRINT COUNT
40 NEXT COUNT
```

```
10 REM Printing lots of letters.
20 FOR NUMTIMES=1 TO 400
30 PRINT "Z-";
40 NEXT NUMTIMES
50 PRINT "WOOPS! TIME TO WAKE UP."
```

```
10 REM Using variables and limits
20 START=1
30 FOR NUM=START TO 10
40 PRINT NUM
50 NEXT NUM
```

```
10 REM Dangerous and improper
15 REM use of a variable
20 FOR COUNTER=1 TO 25
30 COUNTER=COUNTER+1
40 PRINT COUNTER
50 NEXT COUNTER
```

```
10 REM Illustration of a Delay Loop
20 PRINT "Please wait a moment."
30 FOR WAIT=1 TO 3000:NEXT WAIT
40 PRINT "Thank you for waiting."
```

LOOPS
FOR...NEXT STEP

Activity #1

Type in the program and run it.

```
10 REM Illustrating STEP
20 FOR COUNT=1 TO 20 STEP 2
30 PRINT COUNT
40 NEXT COUNT
```

Be sure campers understand what STEP does. Then ask them to experiment with changing the value used with STEP to other numbers.

Add line 15 and change line 20.

```
15 NUMSTEP=4
20 FOR COUNT=1 TO 40 STEP NUM
```

Activity #2

In this activity, campers will examine a program using INPUT to assign values to the limits of the FOR...NEXT loop and to STEP. Enter the following and run the program.

```
 10 REM Using variables with STEP and FOR...NEXT
 20 PRINT "By what number would you like to"
 30 PRINT "count?";
 40 INPUT COUNTBY
 50 PRINT "Let me think..."
 60 FOR WAIT=1 TO 3000:NEXT WAIT
 70 PRINT "OK.  I will count by ";COUNTBY
 80 FOR COUNTER=0 TO 50 STEP COUNTBY
 90 PRINT COUNTER
100 NEXT COUNTER
```

Discuss the program and then challenge campers to write a program that will allow the user to input not only the STEP value, but also the limits of the FOR...NEXT loop. They should ask for a starting number and an ending number and call those values "START" and "FINISH" respectively.

FOR...NEXT STEP
(CONTINUED)

<u>Activity #3</u>

Type in and run the following program. The ESC CTRL-CLEAR is used in the program to clear the screen (Line 20).

```
10 REM A program to count backwards
20 PRINT "↰":REM Clears screen
30 "***COUNTING BACKWARDS***":PRINT
40 PRINT "Please give me a number"
50 PRINT "between 2 and 100.  ";
60 INPUT NUM:PRINT
70 PRINT "Counting backwards can be fun."
80 PRINT "Starting with ";NUM;" and ending with
   1.":PRINT
90 FOR WAIT=1 TO 1000:NEXT WAIT:REM Delay loop to
   allow time to read
100 FOR COUNTER=NUM TO 1 STEP -1
110 PRINT COUNTER;" ";
120 NEXT COUNTER
130 PRINT :PRINT:REM Advances before starting the
    program again.
140 GOTO 40:REM Ask for another input
```

Discuss the FOR...NEXT loop in lines 100 to 120. Have campers change the value used with STEP to reinforce its effect in the program.

<u>Activity #4</u>

The following program gives an excellent demonstration of incrementing and decrementing steps. Type it in and run it.

```
10 REM Sound demo of STEP
20 FOR PITCH=0 TO 255 STEP 1
30 SOUND 0,PITCH,10,10
40 NEXT PITCH
50 FOR PITCH=255 TO 0 STEP -1
60 SOUND 0,PITCH,10,10
70 NEXT PITCH
```

Run the program. Ask campers to change the values used with STEP in both lines 20 and 50 and listen to the effect.

LOOPS
FOR...NEXT STEP
(CAMPER COPY)

```
10 REM Illustrating STEP
20 FOR COUNT=1 TO 20 STEP 2
30 PRINT COUNT
40 NEXT COUNT
```

Add line 15 and change line 20.

```
15 NUMSTEP=4
20 FOR COUNT=1 TO 40 STEP NUM
```

```
 10 REM Using variables with STEP and FOR...NEXT
 20 PRINT "By what number would you like to"
 30 PRINT "count?";
 40 INPUT COUNTBY
 50 PRINT "Let me think..."
 60 FOR WAIT=1 TO 3000:NEXT WAIT
 70 PRINT "OK.  I will count by ";COUNTBY
 80 FOR COUNTER=0 TO 50 STEP COUNTBY
 90 PRINT COUNTER
100 NEXT COUNTER
```

```
10 REM A program to count backwards
20 PRINT "↰":REM Clears screen
30 "***COUNTING BACKWARDS***":PRINT
40 PRINT "Please give me a number"
50 PRINT "between 2 and 100.  ";
60 INPUT NUM:PRINT
70 PRINT "Counting backwards can be fun."
80 PRINT "Starting with ";NUM;" and ending with
   1.":PRINT
90 FOR WAIT=1 TO 1000:NEXT WAIT:REM Delay loop to
   allow time to read
100 FOR COUNTER=NUM TO 1 STEP -1
110 PRINT COUNTER;" ";
120 NEXT COUNTER
130 PRINT :PRINT:REM Advances before starting the
    program again.
140 GOTO 40:REM Ask for another input
```

```
10 REM Sound demo of STEP
20 FOR PITCH=0 TO 255 STEP 1
30 SOUND 0,PITCH,10,10
40 NEXT PITCH
50 FOR PITCH=255 TO 0 STEP -1
60 SOUND 0,PITCH,10,10
70 NEXT PITCH
```

NESTED LOOPS

Nested loops may be quite difficult for some campers to understand. The activities in this lesson are the minimum that should be done.

Activity #1

Type in the following program and then run it.

```
10 REM Demonstration of nested loops.
20 FOR OUTERLOOP=1 TO 5
30 PRINT "OUTERLOOP = ";OUTERLOOP
40 FOR INNERLOOP=1 TO 3
50 PRINT "    INNERLOOP = ";INNERLOOP
60 NEXT INNERLOOP
70 PRINT
80 NEXT OUTERLOOP
```

This program should graphically illustrate how nested FOR...NEXT loops work. Ask campers to change the values in lines 20 and 40 to:

```
20 FOR OUTERLOOP=1 TO 3
40 FOR INNERLOOP=1 TO 5
```

and then predict what the output will be before they run the program.

Activity #2

Encourage campers to predict the output of this program before they run it. Type in:

```
10 REM Printing stars
20 FOR NUMDOWN=1 TO 10
30 FOR NUMACROSS=1 TO 10
40 PRINT "*";
50 NEXT NUMACROSS
60 PRINT
70 NEXT NUMDOWN
```

Call attention to the purpose of the semicolon in line 40 and the PRINT statement in line 60. Change the values in lines 20 and 30, so that the program makes one of the boxes below:

```
**          **********          ********
**          **********          ********
**          **********          ********
**                              ********
**                              ********
**                              ********
```

NESTED LOOPS
(CONTINUED)

Activity #3

Be sure the use of variables and the nested loops in this program are clear.

```
10 REM A triangle of stars
20 FOR ROW=1 TO 10
30 FOR STARS=1 TO ROW
40 PRINT "*";
50 NEXT STARS
60 PRINT
70 NEXT ROW
```

Challenge campers to change the program, so that the triangle is turned upside down like this:

```
*******
******
*****
****
***
**
*
```

Activity #4

A FOR...NEXT loop in this program simulates a tab function. Some of the challenges at the end of the module require that campers understand how the program works. The ESC CTRL-CLEAR (PRINT "↰") is used in the program to clear the screen. Have campers type in the program and run it.

```
10 REM Moving indenting before printing.
20 PRINT "↰":REM Clear screen.
30 PRINT "INDENT"
40 FOR INDENT=1 TO 10
50 FOR SPACES=1 TO INDENT
60 PRINT " ";
70 NEXT SPACES
80 PRINT "INDENT"
90 NEXT INDENT
```

Discuss the relationship between the variables "INDENT" and "SPACES". Ask campers to change the program so that the word "INDENT" moves all the way across and down (but not off) the screen.

NESTED LOOPS
(CONTINUED)

Activity #5

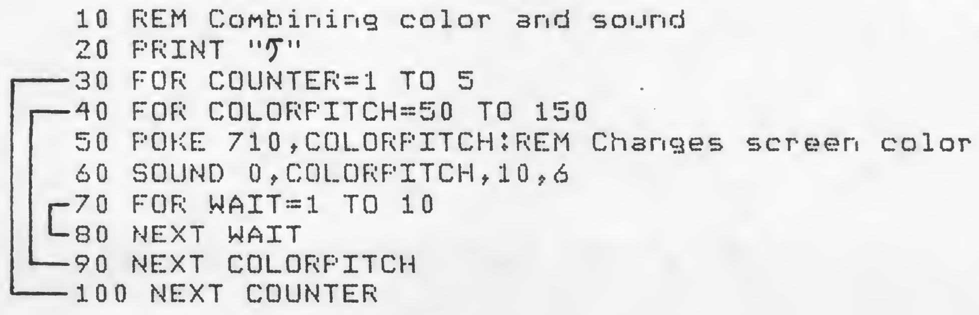
This program combines color and sound using three FOR...NEXT loops. It also uses the ESC CTRL-CLEAR key combination to clear the screen before the program is run. On the screen the character looks like this "↰".

Ask students to type in the program and then show them how to "diagram" the nested loops in order to visually separate them. The "diagramming" is shown below.

```
 10 REM Combining color and sound
 20 PRINT "↰":REM Clears screen
 30 FOR COUNTER=1 TO 5
 40 FOR COLORPITCH=50 TO 150
 50 POKE 710,COLORPITCH:REM Changes screen color
 60 SOUND 0,COLORPITCH,10,6
 70 FOR WAIT=1 TO 10
 80 NEXT WAIT
 90 NEXT COLORPITCH
100 NEXT COUNTER
```

Explain how the variables, "COUNTER", "COLORPITCH", and "WAIT" are used. Run the program. Ask campers to experiment by changing the values in lines 30, 40, and 70 to create different effects.

DIAGRAMMING EXAMPLE



```
10 REM Combining color and sound
20 PRINT "↰"
30 FOR COUNTER=1 TO 5
40 FOR COLORPITCH=50 TO 150
50 POKE 710,COLORPITCH:REM Changes screen color
60 SOUND 0,COLORPITCH,10,6
70 FOR WAIT=1 TO 10
80 NEXT WAIT
90 NEXT COLORPITCH
100 NEXT COUNTER
```

If the lines cross, the loops are not nested properly.

NESTED LOOPS
(CAMPER COPY)

```
10 REM Demonstration of nested loops.
20 FOR OUTERLOOP=1 TO 5
30 PRINT "OUTERLOOP = ";OUTERLOOP
40 FOR INNERLOOP=1 TO 3
50 PRINT "    INNERLOOP = ";INNERLOOP
60 NEXT INNERLOOP
70 PRINT
80 NEXT OUTERLOOP
```

Change the values in lines 20 and 40 to:

```
20 FOR OUTERLOOP=1 TO 3
40 FOR INNERLOOP=1 TO 5
```

and then predict what the output will be before you run the program.

```
10 REM Printing stars
20 FOR NUMDOWN=1 TO 10
30 FOR NUMACROSS=1 TO 10
40 PRINT "*";
50 NEXT NUMACROSS
60 PRINT
70 NEXT NUMDOWN
```

Change the values in lines 20 and 30, so that the program makes one of the boxes below:

```
**         **********          ********
**         **********          ********
**         **********          ********
**                             ********
**                             ********
**                             ********
```

NESTED LOOPS
(CAMPER COPY - CONTINUED)

```
10 REM A triangle of stars
20 FOR ROW=1 TO 10
30 FOR STARS=1 TO ROW
40 PRINT "*";
50 NEXT STARS
60 PRINT
70 NEXT ROW
```

Change the program, so that the triangle is turned upside down like this:

```
*******
******
*****
****
***
**
*
```

```
 10 REM Moving indenting before printing.
 20 PRINT "↰":REM Clear screen.
 30 PRINT "INDENT"
 40 FOR INDENT=1 TO 10
 50 FOR SPACES=1 TO INDENT
 60 PRINT " ";
 70 NEXT SPACES
 80 PRINT "INDENT"
 90 NEXT INDENT
```

```
 10 REM Combining color and sound
 20 PRINT "↰":REM Clear screen
 30 FOR COUNTER=1 TO 5
 40 FOR COLORPITCH=50 TO 150
 50 POKE 710,COLORPITCH:REM Changes screen color
 60 SOUND 0,COLORPITCH,10,6
 70 FOR WAIT=1 TO 10
 80 NEXT WAIT
 90 NEXT COLORPITCH
100 NEXT COUNTER
```

GOTO

Campers have seen GOTO used in previous programs, even though no formal instruction has been done.

Activity #1

Campers may enjoy fooling their parents with this program. The loop should be stopped at Week# 34, since the numbers get too large after that week. Type it in and run it.

```
10 REM The Allowance Con
20 REM Ask for an allowance in a
30 REM different way. Ask for 1 cent
40 REM the first week. Then ask that
50 REM the amount be doubled each week.
60 REM The program shows you how much you
70 REM would earn each week.
80 PRINT " ":REM Clear screen
90 WEEK=1:ALLOWANCE=1
100 PRINT "WEEK #";WEEK
110 PRINT "ALLOWANCE = $";ALLOWANCE/100
120 PRINT
130 ALLOWANCE=ALLOWANCE*2
150 WEEK=WEEK+1
210 GOTO 100
```

Discuss why the output changes after Week #34.

GOTO
(CAMPER COPY)

```
10 REM The Allowance Con
20 REM Ask for an allowance in a
30 REM different way.  Ask for 1 cent
40 REM the first week.  Then ask that
50 REM the amount be doubled each week.
60 REM The program shows you how much you
70 REM would earn each week.
80 PRINT " ":REM Clear screen
90 WEEK=1:ALLOWANCE=1
100 PRINT "WEEK #";WEEK
110 PRINT "ALLOWANCE = $";ALLOWANCE/100
120 PRINT
130 ALLOWANCE=ALLOWANCE*2
150 WEEK=WEEK+1
210 GOTO 100
```

CHALLENGES

Use FOR..NEXT loops to write a program (or programs) to solve one or more of the following problems.

1. Write a program to output one of the designs below. Print your name instead of the word NAME if you do the first one.

```
    A. NAME
        NAME
         NAME
          NAME
           NAME
            NAME
             NAME
```

```
    B.         #
              ###
             #####
            #######
           #########
          ###########
```

```
    C.         *
              * *
             *   *
            *     *
             *   *
              * *
               *
```

```
    D.    XXXXX                   XXXXX
              XXXXX           XXXXX
                  XXXXX   XXXXX
                      XXXXX
                  XXXXX   XXXXX
              XXXXX           XXXXX
          XXXXX                   XXXXX
```

CHALLENGES
(CONTINUED)

2. Write a program to print one of the number sequences.

A. First sequence:

```
5    24    43    62    81    100
```

B. Second sequence:

```
3    40    77    114   151   188
```

3. Write a program that shows all the numbers between 1 and 100 that are divisible by 3. If you have time, change the program, so that a person can ask for all the numbers between two numbers that are divisible by a third number.

MODULE #5 - SUBROUTINES AND STRUCTURED PROGRAMMING

OBJECTIVES

Know what a subroutine is.

Know when it is appropriate to use subroutines.

Know how subroutines work in a program and be able to use subroutines properly.

Recognize the importance of planning before starting to write a program.

Be able to use the top down approach to programming.

Create a library of useful subroutines that can be used in different programs.

MATERIALS REQUIRED

BASIC Cartridge
Camper's Personal Diskette
BASIC Utility Disk:

ANARROW	INTRO.TXT
CHOMP	LEAVING.TXT
MSHUT.GR	BEACH.TXT
MOPEN.GR	STORM.TXT
BOAT.GR	RAIN.TXT
TREE.GR	PHONHOME.TXT
CHMAIN	ENDROPE.TXT
SUN.GR	SUNSHINE.TXT
ROCKET.GR	

REFERENCES

"Slicing Through Spaghetti Code" - Reprint of an article by Art Luehrmann.

Inside Atari Basic - pp. 57-59

Your Atari Computer - pp. 80-81, 86-89

Atari 400/800 Basic Reference Manual - pp. 15-16

Campers must have mastered skills covered in Modules 2-4 before they begin this one. When the module is completed, campers will have written a program that tells a story about a person who goes on a vacation. A BASIC Cartridge and the Camper's Personal Diskette are necessary for each lesson. The prerequisite statements for each lesson are all of the ones covered in the first four modules.

<u>Lesson 1 - Subroutines - An Introduction</u>

Pages 1-6

New Statements
GOSUB ENTER LIST

Materials
BASIC Utility Disk: ANARROW

Uses sample programs to show how subroutines work. Shows how to LIST part of a program to the disk to store it as a subroutine and then recall it using ENTER.

<u>Lesson 2 - Practice Using Subroutines</u>

Pages 7-10

Materials
BASIC Utility Disk:
CHOMP
MSHUT.GR
MOPEN.GR
BOAT.GR
TREE.GR
ROCKET.GR
CHMAIN

Six subroutines are combined to show pictures on the screen. Campers retrieve subroutines from the BASIC Utility Disk, store them on their Personal Diskette, and then practice using their newly created subroutine library.

Lesson 3 - Graphics Subroutines

Pages 11-16

Materials
BASIC Utility Disk: SUN.GR

Campers type in and store several subroutines that will be used in Lesson 5. A format is established for organizing line numbers and names of subroutines. DO NOT SKIP THIS LESSON.

Lesson 4 - Sound Subroutines

Pages 17-23

Campers experiment with SOUND to make sound effects. Four sound effects are stored as subroutines to be used in the project in Lesson 5. DO NOT SKIP THIS LESSON.

Lesson 5 - Writing a Program

Pages 24-38

Materials
BASIC Utility Disk:
- INTRO.TXT
- LEAVING.TXT
- BEACH.TXT
- STORM.TXT
- RAIN.TXT
- PHONHOME.TXT
- ENDROPE.TXT
- SUNSHINE.TXT

"Top Down" programming and proper use of subroutines are stressed. Campers produce a project that tells a story using text, graphics, and sound.

SUBROUTINES
AN INTRODUCTION

Campers should be familiar with Atari graphics and sound capabilities, and they should be able to use variables and loops before beginning these activities. If your group has not completed activities from the first four modules, you should read this module's information cover sheet carefully to determine whether additional instruction is necessary before you start these lessons.

When they have completed this lesson, they should be able to answer the following questions.

1. What is a subroutine?

2. What is the form or structure of a subroutine?

3. Why use subroutines?

4. When is it appropriate to use a subroutine?

5. How does a subroutine work within a program?

It is important that you thoroughly discuss answers to these questions as you teach this lesson and others in the module. Do not merely give campers the code to work with. Subsequent modules presume the ability to use subroutines, and they emphasize structured programming.

<u>Activity #1</u>

Type in the following program.

```
10 REM *  GOSUB Demo  *
20 PRINT "Hi ";
30 GOSUB 100
40 PRINT "Camper!"
50 END
100 REM  *******Sample Subroutine******
110 PRINT "there ";
120 RETURN
```

Run the program. Discuss the subroutine including the following:

1. The subroutine has a REM as the first line to identify it. (Line 100)
2. RETURN must be at the end. This sends the computer back to the main program.
3. The instructions carried out by the subroutine are in the lines between the REM and the RETURN.

SUBROUTRINES
AN INTRODUCTION
(CONTINUED)

4. In the main program, the command, GOSUB, and and a line number tell the computer to go to the specified line in the program, which is the beginning of the subroutine.
5. When the computer returns to the main program from the subroutine, it continues on the next line after the GOSUB command.
6. The main program must have an END statement. Consider what would happen in this simple program if line 50 were not there.
7. REMarks are essential when using subroutines.

Activity #2

Be sure the disk drive, printer, and interface (if you have one) are on before you begin this activity.

The program called ANARROW on the BASIC Utility Disk combines nested FOR..NEXT loops and a subroutine to produce an "animated arrow." Have campers run it from the disk using RUN"D:ANARROW" and then list it to the printer using LIST"P:". A transparency and individual student worksheet accompany this activity in case you cannot use the printer.

Campers should use the hard copy from the printer to "diagram" the loops in the program as shown in your copy of the code. (This diagramming activity was introduced in Module 4 in the lesson called "NESTED LOOPS".)

When you are sure the nested loops are understood, discuss the following:

1. The GOSUB commands in lines 90 and 120, and the END in line 150.
2. The asterisks in lines 10, 70, 100, and 150. Point out the fact that they make it easier to find the REMarks, and thus to read the code. (Consider the subroutine REMark separately)
3. Discuss the REMarks in lines 10800, and 10890-10940. The format of using asterisks to separate subroutines is used throughout the module.
4. Follow the program step by step to see how the subroutine is used. Discuss how the subroutine saved the programmer time.

SUBROUTINES
AN INTRODUCTION
(CONTINUED)

<u>Activity #3</u>

<u>Be sure the disk drive, printer, and interface (if you have one) are on before you begin this activity.</u>

In this activity, campers will save part of the ANARROW program as a subroutine and use it in a simple program they type in. The steps below must be followed exactly.

1. If the program is not in memory, have campers load it from the BASIC Utility Disk using the command LOAD"D:ANARROW". LIST the program to see that it loaded properly.

2. Use the printed copy of the code to find the number of the first line and the last line of the subroutine (10800 and 10890).

3. Remove the BASIC Utility Disk and insert the camper's personal disk.

4. Type LIST"D:ARROW.GR",10800,10890. Explain that this stores the subroutine on their disk in a special way, so that it can be used as a subroutine in another program without having to type it in again. ARROW.GR is the name they will use to retrieve the subroutine. It is like the name they use when they save a program. The .GR indicates to the camper (not to the computer) that this is a graphics subroutine. Do not omit the .GR extension. It will be used in the next lesson. The numbers are the beginning and ending lines that they want to save from the program in memory.

5. Type NEW to clear memory and then LIST to be sure nothing is there. Now type ENTER"D:ARROW.GR". After the disk activity is finished, type LIST. The subroutine is now in memory.

SUBROUTINES
AN INTRODUCTION
(CONTINUED)

6. Type NEW. Ask campers to tell you as a group how they would write a simple program that draws one arrow on the screen. They will need to specify graphics mode, color, and X and Y coordinates for the PLOT statement in the subroutine. Be sure that they include REMarks. Have them type in the lines of the program they write. A sample program is shown below.

```
10 REM *  Drawing an arrow  *
20 GRAPHICS 7+16
40 COLOR 1:REM *  Arrow's color  *
50 GOSUB 10800
60 REM *Line 70 keeps arrow on screen*
70 FOR WAIT=1 TO 1000:NEXT WAIT
80 END
```

7. Now type ENTER"D:ARROW.GR" and LIST the program. Point out that the ENTER command inserts the subroutine into the program from the disk.

8. Remind campers that if they ever have need of an arrow in another program, they now have one available. In a sense, they have taught the computer to draw an arrow. By using subroutines properly, they can "teach" the computer to do many tasks that they can use over and over.

ANIMATED ARROW

(TEACHER COPY)

```
10 REM *  ANIMATED ARROW  *
20 GRAPHICS 7+16
30 Y=40
40 FOR HUE=0 TO 15
50 FOR X=15 TO 105 STEP 5
60 SETCOLOR 0,HUE,2*X/15
70 REM *  DRAW ARROW  *
80 COLOR 1
90 GOSUB 10800
100 REM *  ERASE ARROW  *
110 COLOR 0
120 GOSUB 10800
130 NEXT X
140 NEXT HUE
150 END
10800 REM ******Makes Arrow*********
10810 PLOT X,Y
10815 DRAWTO X-15,Y
10820 DRAWTO X-15,Y+2
10830 DRAWTO X,Y+2
10840 PLOT X+2,Y+1
10850 DRAWTO X-6,Y-3
10860 PLOT X+2,Y+1
10870 DRAWTO X-6,Y+5
10880 RETURN
```

ANIMATED ARROW
(CAMPER COPY)

```
10 REM *  ANIMATED ARROW  *
20 GRAPHICS 7+16
30 Y=40
40 FOR HUE=0 TO 15
50 FOR X=15 TO 105 STEP 5
60 SETCOLOR 0,HUE,2*X/15
70 REM *  DRAW ARROW  *
80 COLOR 1
90 GOSUB 10800
100 REM *  ERASE ARROW  *
110 COLOR 0
120 GOSUB 10800
130 NEXT X
140 NEXT HUE
150 END
10800 REM ******Makes Arrow*********
10810 PLOT X,Y
10815 DRAWTO X-15,Y
10820 DRAWTO X-15,Y+2
10830 DRAWTO X,Y+2
10840 PLOT X+2,Y+1
10850 DRAWTO X-6,Y-3
10860 PLOT X+2,Y+1
10870 DRAWTO X-6,Y+5
10880 RETURN
```

PRACTICE USING SUBROUTINES

Before campers type in and store their own subroutines, this practice lesson should be completed to be sure that they understand how to store and retrieve subroutines. In this lesson six subroutines are combined to show graphics on the screen. Because the graphics characters do not print on the printer, no copy of the code is available for use by campers. They do not, however, have to type anything in. The object of this lesson is to practice using subroutines, not writing them. There is a copy of the code for the teacher.

<u>Be sure the disk drive, printer, and interface (if you have one) are on before you begin this lesson.</u>

1. Have campers run the program on the BASIC Utility Disk called CHOMP to see the output of the program they will be assembling.

2. Type NEW. Enter the subroutine called ROCKET.GR into memory using the ENTER"D:_______" command, putting ROCKET.GR in the blank.

3. The camper's personal disk should be put in the drive and the subroutine stored on it by typing LIST"D:ROCKET.GR".

4. Use the same steps (listed below) to store the other subroutines.

<u>STEPS</u>

1. Type NEW.
2. Put in the BASIC Utility Disk and type ENTER"D:_________".
3. Put in the camper's disk and type LIST"D:_________".

<u>SUBROUTINE NAMES TO USE</u>

MSHUT.GR
MOPEN.GR
BOAT.GR
TREE.GR
CHMAIN (This is the main program.)

PRACTICE USING SUBROUTINES
(CONTINUED)

5. Now that all of the subroutines are stored on the camper's personal disk, the program can be put together. To do that, these steps should be followed:

1. Type NEW (<u>ONLY</u> <u>ONCE!</u>)
2. Type ENTER"D:__________" using each of the subroutine names in place of the blank.
3. Run the program to see if everything was done properly.

6. Remind campers that they now have five more graphics subroutines that are at their disposal for use in programs.

7. It may be necessary to explain that the character in line 20 is made with the ESC CTRL-CLEAR key combination. It clears the screen when the program starts.

8. Discuss the fact that the GOTO statement in line 200 takes the place of the END statement we have used.

9. Experiment with the program that they have put together by changing the GOSUBS in lines 110, 130, 150, 170, and 190 of the main program.

CHOMP
(TEACHER'S COPY)

```
10 DIM LINE$(25),L$(25),B$(25)
20 PRINT "}":REM CLEAR SCREEN
30 POKE 752,1:REM TURN OFF CURSOR
100 POSITION 5,5
110 GOSUB 11300:REM TREE
120 POSITION 24,4
130 GOSUB 10900:REM ROCKET
140 POSITION 22,15
150 GOSUB 11200:REM BOAT
160 POSITION 6,16
170 GOSUB 11000:REM SHUT MOUTH
180 POSITION 6,16
190 GOSUB 11100:REM OPEN MOUTH
200 GOTO 160
10900 REM *********ROCKET***********
10910 LINE$=""
10920 PRINT "       ";LINE$;
10930 PRINT "   |   ";LINE$;
10940 PRINT "
              ";LINE$;
10950 PRINT "      ";LINE$;
10960 PRINT "      ";LINE$;
10970 PRINT "      ";LINE$;
10980 PRINT "
              ";LINE$;
10985 PRINT " /`\ ";LINE$;
10990 PRINT "       ";
10995 RETURN
11000 REM *******SHUT MOUTH**********
11010 LINE$=""
11020 PRINT "       ";LINE$;
11030 PRINT "
              ";LINE$;
11040 PRINT "     ";LINE$;
11050 PRINT "    ";LINE$;
11060 PRINT "
              ◥;◢INE$;
11070 PRINT "
              ";LINE$;
11080 PRINT "       ";
11090 RETURN
```

```
11100 REM ********OPEN MOUTH*********
11110 LINE$=""
11120 PRINT "       ";LINE$;
11130 PRINT "
         ";LINE$;
11140 PRINT "      ";LINE$;
11150 PRINT "    ";LINE$;
11160 PRINT "

          ";LINE$;
11170 PRINT "
         ";LINE$;
11180 PRINT "       ";
11190 RETURN
11200 REM **********BOAT*************
11210 LINE$=""
11220 PRINT "          ";LINE$;
11230 PRINT "    |     ";LINE$;
11240 PRINT "    |
              ";LINE$;
11250 PRINT "
     ";LINE$;
11260 PRINT "
            ";LINE$;
11270 PRINT "
        ";LINE$;
11280 PRINT "
            ";LINE$;
11290 PRINT "          ";
11295 RETURN
11300 REM ***********TREE************
11310 LINE$=""
11320 PRINT "       ";LINE$;
11330 PRINT "   ^   ";LINE$;
11340 PRINT "
         ";LINE$;
11350 PRINT "
         ";LINE$;
11360 PRINT "
         ";LINE$;
11370 PRINT "
         ";LINE$;
11380 PRINT "       ";LINE$;
11390 PRINT "       ";
11395 RETURN
```

GRAPHICS SUBROUTINES

This lesson has two purposes. One is to encourage the idea of creating an organized "library" of subroutines that will enable the camper to save time in writing programs. The other is to type in and save several subroutines that will be used in the final lesson in this module. DO NOT SKIP THIS LESSON.

All subroutines that will go into the "library" will begin with a REM as identification. It is good practice to use an identifier that explains what the subroutine does. Graphics subroutines will use line numbers beginning with 10,000. The REM that contains the identifier is on a number that is an increment of 100. The file names have a .GR extension to indicate graphics.

Stress the necessity of typing NEW each time a new subroutine is typed in and stored.

Be sure the disk drive, printer, and interface (if you have one) are on before you begin this lesson.

Activity #1

This subroutine plots points to make a circle.

Type NEW before beginning.

1. Type in the following:

```
10000 REM *****Going in circles*****
10010 GRAPHICS 7+16:COLOR 2
10020 FOR COUNTER=1 TO 50
10030 Z=Z+0.5
10040 X=SIN(Z)*25:Y=COS(Z)*22
10050 PLOT X+80,Y+45
10060 NEXT COUNTER
10070 RETURN
```

2. Store the subroutine on the camper's personal disk by typing LIST"D:CIRCLE1.GR". The name, CIRCLE1.GR, must be used.

3. Type NEW to clear memory and then type ENTER"D:CIRCLE1.GR" to put the subroutine back in memory. Type LIST to see that it is there.

4. To see what the subroutine does, campers should type in the two line program below.

```
10 GOSUB 10000
20 END
```

This program will be used to test all the subroutines in this and the next lesson.

Activity #2

As stated in other lessons, delay loops are very tricky in BASIC. A FOR..NEXT loop at the top of a program executes much faster than one near the bottom. Timing can be made more consistent by putting the loop in a subroutine. In this activity, campers will store a delay loop and then use it in another subroutine. Talk about nested subroutines and the dangers of nesting that is too deep.

1. Type in:

```
NEW

29000 REM *********Wait Loop**********
29010 FOR WAIT=1 TO 500:NEXT WAIT
29020 RETURN
```

and store it using LIST"D:WAIT.LP".

2. Type NEW and then the following subroutine. Campers should put their name where it says "Your Name"

```
10100 REM *******Author Page*********
10110 GRAPHICS 2+16
10120 POSITION 4,2:PRINT #6;"*************"
10130 POSITION 4,3:PRINT #6;"*           *"
10135 POSITION 4,4:PRINT #6;"*     BY    *"
10140 POSITION 4,5:PRINT #6;"*           *"
10150 POSITION 4,6:PRINT #6;"* Your  Name *"
10155 POSITION 4,7:PRINT #6;"*           *"
10160 POSITION 4,8:PRINT #6;"*************"
10170 RETURN
```

GRAPHICS SUBROUTINES
(CONTINUED)

3. In order to see what this subroutine does, campers will need to enter the delay loop subroutine. They should type ENTER"D:WAIT.LP" and then use the three line program:

```
10 GOSUB 10100
20 GOSUB 29000
30 END
```

Have campers delete line 20 and run the program.

Activity #3

Campers must type in and store two more subroutines to be used in the last lesson. They can do this on their own if you feel that they are ready. However, stress the fact that the subroutines must be typed in exactly as shown and the name used to store them must be the one given. Remind them to type NEW before starting the second subroutine.

Type NEW before you begin.

TITLE1.GR

```
10300 REM ********Title page********
10310 GRAPHICS 2+16:COLOR 2
10315 PRINT #6:PRINT #6:PRINT #6:PRINT #6
10320 PRINT #6;"         A"
10325 PRINT #6;"      VACATION"
10330 PRINT #6;"       STORY"
10340 PLOT 1,1
10345 DRAWTO 19,1
10350 DRAWTO 19,9
10355 DRAWTO 1,9
10360 DRAWTO 1,1
10370 RETURN
```

Type LIST"D:TITLE1.GR" to store the subroutine.

GRAPHICS SUBROUTINES
(CONTINUED)

Type NEW before you begin.

RAIN.GR

```
10600 REM **********Rain**************
10605 FOR LOOP=1 TO 3
10610 GRAPHICS 3+16
10620 FOR COUNTER=1 TO 84
10630 PRINT #6,"+";
10640 NEXT COUNTER
10650 GRAPHICS 0
10660 NEXT LOOP
10670 RETURN
```

Type LIST"D:RAIN.GR" to store the subroutine.

4. A subroutine called SUN.GR will be needed. Since it would be extremely time consuming to type it in, have campers ENTER it from the BASIC Utility Disk (ENTER"D:SUN.GR") and then store it on their personal disk using LIST"D:SUN.GR". To see the sun, type:

```
ENTER"D:WAIT.LP"
10 GOSUB 10200
20 GOSUB 29000
30 END
```

and then run the program. A copy of the subroutine code is included for the teacher's use.

```
10200 REM ***********Sun***********
10205 GRAPHICS 3+16
10210 ? #6;"                =     "
10215 ? #6;"      =        =      "
10220 ? #6;"       =      =       "
10225 ? #6;"        =    =        "
10230 ? #6;"         ====         "
10235 ? #6;"        ======        "
10240 ? #6;"       ======== = = = "
10245 ? #6;" = = = ========       "
10250 ? #6;"       ========       "
10255 ? #6;"        ======        "
10260 ? #6;"         ==== =       "
10270 ? #6;"         =     =      "
10275 ? #6;"        =       =     "
10280 ? #6;"       =         =    "
10285 ? #6;"      =               "
10290 RETURN
```

GRAPHICS SUBROUTINES
CAMPER COPY

Type NEW before beginning.

```
10000 REM *****Going in circles*****
10010 GRAPHICS 7+16:COLOR 2
10020 FOR COUNTER=1 TO 50
10030 Z=Z+0.5
10040 X=SIN(Z)*25:Y=COS(Z)*22
10050 PLOT X+80,Y+45
10060 NEXT COUNTER
10070 RETURN
```

Store it using LIST"D:CIRCLE1.GR

Type NEW before beginning.

```
29000 REM **********Wait Loop**********
29010 FOR WAIT=1 TO 500:NEXT WAIT
29020 RETURN
```

Store it using LIST"D:WAIT.LP".

```
10100 REM *******Author Page*********
10110 GRAPHICS 2+16
10120 POSITION 4,2:PRINT #6;"**************"
10130 POSITION 4,3:PRINT #6;"*            *"
10135 POSITION 4,4:PRINT #6;"*     BY     *"
10140 POSITION 4,5:PRINT #6;"*            *"
10150 POSITION 4,6:PRINT #6;"* Your  Name *"
10155 POSITION 4,7:PRINT #6;"*            *"
10160 POSITION 4,8:PRINT #6;"**************"
10170 RETURN
```

Store it using LIST"D:AUTHOR.GR".

GRAPHICS SUBROUTINES
CAMPER COPY

Type NEW before you begin.

TITLE1.GR

```
10300 REM ********Title page********
10310 GRAPHICS 2+16:COLOR 2
10315 PRINT #6:PRINT #6:PRINT #6:PRINT #6
10320 PRINT #6;"         A"
10325 PRINT #6;"      VACATION"
10330 PRINT #6;"       STORY"
10340 PLOT 1,1
10345 DRAWTO 19,1
10350 DRAWTO 19,9
10355 DRAWTO 1,9
10360 DRAWTO 1,1
10370 RETURN
```

Type LIST"D:TITLE1.GR" to store the subroutine.

Type NEW before you begin.

RAIN.GR

```
10600 REM *********Rain**************
10605 FOR LOOP=1 TO 3
10610 GRAPHICS 3+16
10620 FOR COUNTER=1 TO 84
10630 PRINT #6,"+";
10640 NEXT COUNTER
10650 GRAPHICS 0
10660 NEXT LOOP
10670 RETURN
```

Type LIST"D:RAIN.GR" to store the subroutine.

SOUND SUBROUTINES

This lesson will allow campers to experiment with the SOUND statement to make sound effects. Four of the subroutines will be used in the final project. Be sure that Activity #1 is completed before going on in the module. It is important that the subroutines be stored exactly as directed. The remaining activities provide practice with variables and FOR..NEXT loops. It may be necessary for you to set guidelines for the amount of noise you (and fellow campers) can tolerate.

The format for subroutines begun in the Graphics Subroutines lesson is continued here. Line numbers beginning with 20,000 will be used. The extension .SO is included in the name of the subroutine to identify it as a sound effect.

Remember to stress the necessity of typing NEW each time a new subroutine is typed in and stored.

Be sure the disk drive, printer, and interface (if you have one) are on before you begin this lesson.

Activity #1

These four subroutines must be stored on the camper's personal disk, since they will be used in the next lesson. Since delay loops execute much faster at the beginning of programs than they do at the end, three of the subroutines will have to be changed in order to hear the proper sound if they are used separate from the larger program.

1. Type in the following:

```
20900 REM ***********Ocean***********
20910 FOR LOOP=1 TO 2
20920 FOR PITCH=0 TO 12
20930 SOUND 0,PITCH,8,6
20940 FOR WAIT=1 TO 5:NEXT WAIT
20950 NEXT PITCH
20960 FOR PITCH=12 TO 0 STEP -1
20970 SOUND 0,PITCH,8,4
20975 FOR WAIT=1 TO 17:NEXT WAIT
20980 NEXT PITCH
20985 NEXT LOOP
20990 SOUND 0,0,0,0
20995 RETURN
```

Be sure the camper's personal disk is in the drive. Type LIST"D:OCEAN.SO" to store the subroutine. In order to hear the ocean sound change these lines:

```
20940 FOR WAIT=1 TO 35
20975 FOR WAIT=1 TO 125
```

and add:

```
10 GOSUB 20900
20 GOTO 10
```

Run the program.

2. The next steps use the same sequence used in #1. Type NEW and then:

```
20300 REM ***Telephone Busy Signal***
20305 FOR RINGS=1 TO 9
20310 SOUND 2,40,6,10
20320 FOR WAIT=1 TO 50:NEXT WAIT
20330 SOUND 2,0,0,0
20340 FOR WAIT=1 TO 25:NEXT WAIT
20350 NEXT RINGS
20360 RETURN
```

Type LIST"D:BUSY.SO". Change these lines:

```
20320 FOR WAIT=1 TO 400:NEXT WAIT
20340 FOR WAIT=1 TO 400:NEXT WAIT
```

Add:

```
10 GOSUB 20300
20 GOTO 10
```

and run the program.

SOUND SUBROUTINES
(CONTINUED)

3. Use LIST"D:TRAIN.SO to store this:

```
20500 REM *****Steam Locomotive******
20510 FOR LOOP=1 TO 25
20520 FOR LOUD=10 TO 0 STEP -1
20530 SOUND 0,15,0,LOUD
20540 NEXT LOUD
20550 NEXT LOOP
20560 SOUND 0,0,0,0
20570 RETURN
```

Change this line:

```
20520 FOR LOUD=15 TO 0 STEP -1
```

add:

```
10 GOSUB 20500
20 GOTO 10
```

and run the program.

4. You can hear the sound made by this subroutine without changing any of the lines. Store it using LIST"D:BIRDS.SO".

```
20100 REM *******Chirping Birds******
20110 FOR LOOP=1 TO 4
20120 FOR COUNT=1 TO 5
20130 FOR PITCH=1 TO 15
20140 SOUND 2,PITCH,10,8
20150 NEXT PITCH
20160 NEXT COUNT
20170 NEXT LOOP
20180 SOUND 2,0,0,0
20190 RETURN
```

ACTIVITY #2

Campers may choose to add the subroutines listed in this activity to their "library". Remind them that each name should end with .SO, so that they are identified as sound subroutines. A suggested name is listed at the end of each entry. This can be done as an individual activity if campers are familiar with the process. See the page called "ADDITIONAL SOUND SUBROUTINES" for the subroutine listings. Copies of the code are also available on a transparency.

ADDITIONAL SOUND SUBROUTINES
(CAMPER COPY)

DIRECTIONS

Type in the following subroutines. As you finish each one, store it on your disk using the LIST command and the name given at the end of the listing. Be sure to use the line numbers specified, so that you can include more than one of the subroutines in the same program. Type NEW before you start each sound effect. After storing the sound effect, add a GOSUB and listen to the result.

1. A Siren

```
20000 REM ***** A Siren *****
20010 FOR COUNT=1 TO 20
20020 FOR PITCH=20 TO 50
20030 SOUND 0,PITCH,10,8
20040 NEXT PITCH
20050 NEXT COUNT
20060 SOUND 0,0,0,0
20070 RETURN

LIST"D:SIREN.SO"
```

2. Exploding Bomb

```
20400 REM ***** Exploding Bomb *****
20410 FOR PITCH=30 TO 200
20420 SOUND 0,PITCH,10,8
20430 NEXT PITCH
20440 SOUND 0,80,0,11
20450 FOR WAIT=1 TO 500:NEXT WAIT
20460 SOUND 0,0,0,0
20470 RETURN

LIST"D:BOMB.SO"
```

3. A Bouncing Ball

```
20600 REM ***** Bouncing Ball *****
20610 FOR BOUNCES=1 TO 8
20620 FOR C=1 TO 8
20630 SOUND 0,124,14,4
20640 NEXT C
20650 SOUND 0,0,0,0
20660 FOR WAIT=1 TO 400:NEXT WAIT
20670 NEXT BOUNCES
20680 RETURN

LIST"D:BOUNCE.SO"
```

ADDITIONAL SOUND SUBROUTINES
(CAMPER COPY)

4. A Jackhammer

```
20700 REM ***** Jackhammer *****
20710 FOR HAMMER=1 TO 300
20720 FOUND 0,100,6,4
20730 NEXT HAMMER
20740 SOUND 0,0,0,0
20750 FOR WAIT=1 TO 500:NEXT WAIT
20760 RETURN

LIST"D:JAKHAMR.SO"
```

5. Thunder

```
20800 REM ***** Thunder *****
20810 FOR LOOP=1 TO 4
20820 FOR PITCH=1 TO 255
20830 SOUND 0,PITCH,8,15
20840 NEXT PITCH
20850 SOUND 0,0,0,0
20860 FOR WAIT=1 TO 350:NEXT WAIT
20870 NEXT LOOP
20880 RETURN

LIST"D:THUNDER.SO"
```

6. Argument Between Parent and Child Computers

```
21000 REM **** Computer Argument ****
21010 FOR PARENT=1 TO 100
21020 SOUND 0,INT(RND(0)*25),10,8
21030 NEXT PARENT
21040 SOUND 0,0,0,0
21050 FOR WAIT=1 TO 500:NEXT WAIT
21060 FOR CHILD=1 TO 100
21070 SOUND 1,INT(RND(0)*200),10,8
21080 NEXT CHILD
21085 SOUND 1,0,0,0
21090 FOR WAIT=1 TO 200:NEXT WAIT
21095 RETURN

LIST"D:ARGUE.SO"
```

SOUND SUBROUTINES
(CAMPER COPY)

```
20900 REM ***********Ocean***********
20910 FOR LOOP=1 TO 2
20920 FOR PITCH=0 TO 12
20930 SOUND 0,PITCH,8,6
20940 FOR WAIT=1 TO 5:NEXT WAIT
20950 NEXT PITCH
20960 FOR PITCH=12 TO 0 STEP -1
20970 SOUND 0,PITCH,8,4
20975 FOR WAIT=1 TO 17:NEXT WAIT
20980 NEXT PITCH
20985 NEXT LOOP
20990 SOUND 0,0,0,0
20995 RETURN
```

```
LIST"D:OCEAN.SO"
```

```
20300 REM ***Telephone Busy Signal***
20305 FOR RINGS=1 TO 9
20310 SOUND 2,40,6,10
20320 FOR WAIT=1 TO 50:NEXT WAIT
20330 SOUND 2,0,0,0
20340 FOR WAIT=1 TO 25:NEXT WAIT
20350 NEXT RINGS
20360 RETURN
```

```
LIST"D:BUSY.SO"
```

SOUND SUBROUTINES
(CAMPER COPY)

```
20500 REM *****Steam Locomotive******
20510 FOR LOOP=1 TO 25
20520 FOR LOUD=10 TO 0 STEP -1
20530 SOUND 0,15,0,LOUD
20540 NEXT LOUD
20550 NEXT LOOP
20560 SOUND 0,0,0,0
20570 RETURN
```

```
LIST"D:TRAIN.SO"
```

```
20100 REM *******Chirping Birds******
20110 FOR LOOP=1 TO 4
20120 FOR COUNT=1 TO 5
20130 FOR PITCH=1 TO 15
20140 SOUND 2,PITCH,10,8
20150 NEXT PITCH
20160 NEXT COUNT
20170 NEXT LOOP
20180 SOUND 2,0,0,0
20190 RETURN
```

```
LIST"D:BIRDS.SO"
```

WRITING A PROGRAM

At some time during the presentation of this lesson you should cover the following:

1. Subroutines can make complex problems easier to solve.
2. The importance of planning before starting to write the program code.
3. The importance of documentation in helping to debug a program.
4. How to do top down programming.
5. The time saved by building a subroutine library.
6. How to determine when a section of a program should be written in a subroutine.
7. A review of the way BASIC handles delay loops, and the necessity of using subroutines to make timing more consistent.
8. The proper form of a subroutine, including a REM at the beginning.
9. The use of LIST"D:_______" and ENTER"D:_________" to store and retrieve subroutines. The need to use descriptive names when storing subroutines.
10. The importance of organizing line numbers when storing subroutines, so that when they are entered, they will not overwrite each other.

There are five activities in the lesson. They should be done in order and none should be skipped. The lesson should not be done individually by students, since much of the learning will take place in discussions.

This lesson is the foundation for instruction that will take place in later modules. It is extremely important that all campers who are in Book 3 complete it, even if they have had some experience programming.

WRITING A PROGRAM
(CONTINUED)

<u>Be sure the disk drive, printer, and interface (if you have one) are on before you begin this lesson.</u>

<u>Activity #1</u>

It will be useful for campers to know how to print a disk directory and to list subroutines to the printer.

1. Put the camper's personal disk in the drive and turn the computer on.

2. When the disk activity stops, type DOS.

3. When the directory appears, type "A" and press RETURN twice to see what is on the camper's disk.

4. Type "A" again. When the line:

   ```
   DIRECTORY--SEARCH SPEC, LIST FILE?
   ```

   appears, type:

   ```
   ,P:
   ```

   This will list the directory on the printer.

5. Type "B" to return to BASIC.

6. Choose a subroutine that is on the directory the camper just printed. Put it in memory using the ENTER"D:__________" command. Then type:

   ```
   LIST"P:"
   ```

   to list the subroutine to the printer.

<u>Activity #2</u>

This activity takes students through the steps of planning a program. A copy of the completed program is available for the teacher's use.

1. General description of the problem.

    Write a short story about a person who leaves the city to go to the beach on a vacation.

2. Be more specific about what the program will do by writing out the story.

    A man named Fred was very bored with life. It seemed like all he ever did was go around in circles. One day he decided to leave the big city. He got on a train and went to the beach for a vacation. The day he arrived, it was sunny and warm. The sound of the ocean was very calming to his nerves. However...That night a storm came up and it rained and it rained and it rained. He decided to phone home to see if the weather was any better there. But since his children were always on the phone, all he got was a busy signal. Just as he was at the end of his rope and ready to return home, the birds began to sing, the sun came out, and he....

3. Decide what pictures and sound effects might be appropriate and make a list of them. Then determine which ones are available in the subroutine library and which ones need to be written. In this case, the subroutines have been written in preparation for this activity. Explain that normally, very little code will be available at this stage of planning. This should emphasize the fact that keeping a subroutine library can save a great deal of time.

4. Divide the story into sections that will fit with the sound and graphics subroutines.

WRITING A PROGRAM
(CONTINUED)

5. Write out English statements that show the solution step by step.

> Title screen
> Author screen
> Introduce the main character and his problem
> Graphics routine for going around in circles
> Leaves the city on a train
> Train sound effect
> Arrives at the beach
> Ocean sound effect
> A storm comes up and it starts raining
> Graphics routine for rain
> It rains some more
> Graphics routine for rain
> It rains some more
> Graphics routine for rain
> He decides to phone home to ask about the weather
> Busy signal sound effect
> He is at the end of his rope when birds sing
> Birds chirping sound effect
> The sun comes out
> Graphics routine for sun
> And he...To be continued.

<u>Activity #3</u>

The English sentences are then translated into code and the program is run. More refinement takes place as bugs are found.

1. A skeletal listing of the main program is on the page called, "VACATION - MAIN PROGRAM". Campers should fill in the blanks and write the code for the English statements. This should be done as a group (or you must check to make sure all information is correct). Their completed version will be used in Activity #4 to type in the main program.

WRITING A PROGRAM
(CAMPER COPY)

1. General description of the problem.

   Write a short story about a person who leaves the city to go to the beach on a vacation.

2. Be more specific about what the program will do by writing out the story.

   A man named Fred was very bored with life. It seemed like all he ever did was go around in circles. One day he decided to leave the big city. He got on a train and went to the beach for a vacation. The day he arrived, it was sunny and warm. The sound of the ocean was very calming to his nerves. However...That night a storm came up and it rained and it rained and it rained. He decided to phone home to see if the weather was any better there. But since his children were always on the phone, all he got was a busy signal. Just as he was at the end of his rope and ready to return home, the birds began to sing, the sun came out, and he....

3. Decide what pictures and sound effects might be appropriate and make a list of them. Then determine which ones are available in the subroutine library and which ones need to be written.

4. Divide the story into sections that will fit with the sound and graphics subroutines.

WRITING A PROGRAM
(CAMPER COPY - CONTINUED)

5. Write out English statements that show the solution step by step.

Title screen
Author screen
Introduce the main character and his problem
Graphics routine for going around in circles
Leaves the city on a train
Train sound effect
Arrives at the beach
Ocean sound effect
A storm comes up and it starts raining
Graphics routine for rain
It rains some more
Graphics routine for rain
It rains some more
Graphics routine for rain
He decides to phone home to ask about the weather
Busy signal sound effect
He is at the end of his rope when birds sing
Birds chirping sound effect
The sun comes out
Graphics routine for sun
And he...To be continued.

VACATION — MAIN PROGRAM

```
 10 REM ******** MAIN PROGRAM ********
 50 GOSUB 10300:REM *   Title Page   *
 55 GOSUB 29000:REM *  Wait Loop     *
 60 GOSUB _____:REM *  Author Page   *
 65 GOSUB _____:REM *  Wait Loop     *
 70 GOSUB 1100:REM *  Introduction   *
 80 GOSUB _____:REM *   Wait Loop    *
 90 GOSUB _____:REM *Going in Circles*
100 GOSUB 1200:REM *Leaving the city *
110 GOSUB _____:REM *  Train Sound   *
120 GOSUB 1300:REM * Arrive at beach *
130 GOSUB _____:REM * Ocean Sound    *
140 GOSUB 1400:REM *   The Storm     *
150 GOSUB _____:REM *   Wait Loop    *
160 GOSUB _____:REM *  Rain Graphic  *
170 GOSUB 1700:REM *     Rain text   *
180 GOSUB _____:REM *   Wait Loop    *
190 GOSUB _____:REM *  Rain graphic  *
200 GOSUB 1700:REM *     Rain text   *
210 GOSUB _____:REM *   Wait Loop    *
___ Graphics routine for rain drops
230 GOSUB 1500:REM *   Phone Home    *
___ Sound effect for telephone busy signal
250 GOSUB 1800:REM *  End of Rope    *
___ Sound effect of birds chirping
270 GOSUB 1900:REM *  Sunshine text  *
___ Delay loop to keep text on screen
___ Graphics routine for sun shining
295 GOSUB 29000:REM *   Wait Loop    *
300 GOSUB 2000:REM *  Continued text *
___ Delay loop to keep text on screen.
350 END
360 REM
370 REM
380 REM
```

<u>LIST OF SUBROUTINES</u>

INTRO.TXT
LEAVING.TXT
BEACH.TXT
STORM.TXT
RAIN.TXT
PHONHOME.TXT
ENDROPE.TXT
SUNSHINE.TXT
CONTINUE.TXT

WRITING A PROGRAM
(CONTINUED)

2. The subroutines listed below are on the BASIC Utility Disk. Enter and store them using the following steps exactly as they are written. They put the text on the screen. The information in parentheses is the REMark in the main program.

Use these steps for each subroutine.

1. Type NEW to clear memory.
2. Put in the BASIC Utility Disk.
3. Type ENTER"D:________", putting the subroutine name in the blank.
4. Put in the camper's personal disk.
5. Type LIST"D:________", putting the subroutine name in the blank.
6. GOTO 1.

<u>LIST OF SUBROUTINES</u>

INTRO.TXT
LEAVING.TXT
BEACH.TXT
STORM.TXT
RAIN.TXT
PHONHOME.TXT
ENDROPE.TXT
SUNSHINE.TXT

<u>Activity #4</u>

Putting it all together.

1. Type in the main program.
2. Print a directory of the camper's disk to get a list of subroutines.
3. ENTER all subroutines. <u>DO NOT TYPE NEW AFTER EACH ENTRY!</u>
4. Run the program and debug it.
5. Save the program using SAVE"D:__________", putting a name the camper chooses in the blank.

WRITING A PROGRAM
(CONTINUED)

Activity #5

Complete at least one of the challenges listed below.

1. Finish the story by adding text, sound, and graphics subroutines.

2. Rearrange the main program, so that action happens in a different sequence.

3. Use entries in the subroutine library to write an original program that tells a story.

A VACATION STORY

```
10 REM *********MAIN PROGRAM*********
50 GOSUB 10300:REM *   Title Page   *
55 GOSUB 29000:REM *   Wait Loop    *
60 GOSUB 10100:REM *  Author Page   *
65 GOSUB 29000:REM *   Wait Loop    *
70 GOSUB 1100:REM * Introduction    *
80 GOSUB 29000:REM *   Wait Loop    *
90 GOSUB 10000:REM *Going in circles*
100 GOSUB 1200:REM *Leaving the city *
110 GOSUB 20500:REM *  Train sound   *
120 GOSUB 1300:REM * Arrive at beach *
130 GOSUB 20900:REM * Ocean sound    *
140 GOSUB 1400:REM *   The Storm     *
150 GOSUB 29000:REM *   Wait Loop    *
160 GOSUB 10600:REM *  Rain graphic  *
170 GOSUB 1700:REM *    Rain text    *
180 GOSUB 29000:REM *   Wait Loop    *
190 GOSUB 10600:REM *  Rain graphic  *
200 GOSUB 1700:REM *    Rain text    *
210 GOSUB 29000:REM *   Wait Loop    *
220 GOSUB 10600:REM *  Rain graphic  *
230 GOSUB 1500:REM *   Phone Home    *
240 GOSUB 20300:REM *  Busy Signal   *
250 GOSUB 1800:REM *  End of Rope    *
260 GOSUB 20100:REM * Chirping birds *
270 GOSUB 1900:REM *  Sunshine text  *
280 GOSUB 29000:REM *   Wait Loop    *
290 GOSUB 10200:REM *  Sun graphic   *
295 GOSUB 29000:REM *   Wait Loop    *
300 GOSUB 2000:REM * Continued text  *
310 GOSUB 29000:REM *   Wait Loop    *
350 END :REM * End of Main Program   *
360 REM
370 REM
380 REM
1100 REM ******Introduction**********
1105 GRAPHICS 1+16
1110 ? #6
1115 ? #6
1120 ? #6
1125 ? #6;"A MAN NAMED FRED"
1127 ? #6
1130 ? #6;"WAS VERY BORED"
1135 ? #6
1140 ? #6;"WITH LIFE.  IT"
1145 ? #6
1150 ? #6;"SEEMED LIKE ALL HE"
1155 ? #6
1160 ? #6;"EVER DID WAS GO"
1165 ? #6
1170 ? #6;"AROUND IN CIRCLES."
1180 RETURN
```

A VACATION STORY

```
1200 REM *****Leaving the city*******
1205 GRAPHICS 1+16
1210 ? #6
1215 ? #6
1220 ? #6
1225 ? #6;"ONE DAY HE "
1227 ? #6
1230 ? #6;"DECIDED TO LEAVE"
1235 ? #6
1240 ? #6;"THE BIG CITY."
1245 ? #6
1250 ? #6;"HE GOT ON A"
1255 ? #6
1265 ? #6;"TRAIN......."
1270 RETURN
1300 REM *****Arrival at beach*******
1302 GRAPHICS 1+16
1305 ? #6
1310 ? #6;"......AND WENT TO"
1315 ? #6
1320 ? #6;"THE BEACH FOR A"
1325 ? #6
1330 ? #6;"VACATION. THE DAY"
1335 ? #6
1340 ? #6;"HE ARRIVED, IT WAS"
1345 ? #6
1350 ? #6;"SUNNY AND WARM."
1355 ? #6
1360 ? #6;"THE SOUND OF THE"
1365 ? #6
1370 ? #6;"OCEAN WAS VERY"
1375 ? #6
1380 ? #6;"CALMING TO HIS"
1385 ? #6
1390 ? #6;"NERVES."
1395 RETURN
1400 REM **********Storm*************
1410 GRAPHICS 2+16
1425 ? #6
1440 ? #6
1445 ? #6;"H O W E V E R....."
1450 ? #6
1455 ? #6;"THAT NIGHT A STORM"
1460 ? #6
1465 ? #6;"CAME UP AND IT"
1470 ? #6
1475 ? #6;"RAINED....."
1480 RETURN
```

A VACATION STORY

```
1500 REM ********Phone home**********
1501 GRAPHICS 1+16
1505 ? #6
1510 ? #6;"HE DECIDED TO "
1515 ? #6
1520 ? #6;"PHONE HOME TO SEE"
1525 ? #6
1530 ? #6;"IF THE WEATHER"
1535 ? #6
1540 ? #6;"WAS ANY BETTER "
1545 ? #6
1550 ? #6;"THERE.  BUT SINCE"
1555 ? #6
1560 ? #6;"HIS CHILDREN WERE"
1565 ? #6
1570 ? #6;"ALWAYS ON THE "
1575 ? #6
1580 ? #6;"PHONE, ALL HE GOT"
1585 ? #6
1590 ? #6;"WAS A BUSY SIGNAL."
1595 RETURN
1700 REM ******And it rained*********
1710 GRAPHICS 2+16
1720 ? #6
1730 ? #6
1740 ? #6
1750 ? #6
1755 ? #6
1760 ? #6;"  AND IT RAINED..."
1770 RETURN
1800 REM *****End of his rope********
1805 GRAPHICS 1+16
1810 ? #6
1815 ? #6
1820 ? #6
1825 ? #6
1830 ? #6
1840 ? #6;"JUST AS HE WAS AT"
1845 ? #6
1850 ? #6;"THE END OF HIS "
1855 ? #6
1860 ? #6;"ROPE AND READY TO"
1865 ? #6
1870 ? #6;"RETURN HOME, THE"
1875 ? #6
1880 ? #6;"BIRDS BEGAN TO"
1885 ? #6
1890 ? #6;"SING..."
1895 RETURN
```

A VACATION STORY

```
10200 REM ***********Sun***********
10205 GRAPHICS 3+16
10210 ? #6;"                =     "
10215 ? #6;"      =        =      "
10220 ? #6;"       =      =       "
10225 ? #6;"        =    =        "
10230 ? #6;"         ====         "
10235 ? #6;"        ======        "
10240 ? #6;"       ======== = = = "
10245 ? #6;" = = = ========       "
10250 ? #6;"       ========       "
10255 ? #6;"        ======        "
10260 ? #6;"         ==== =       "
10270 ? #6;"         =     =      "
10275 ? #6;"        =       =     "
10280 ? #6;"       =         =    "
10285 ? #6;"      =               "
10290 RETURN
10300 REM ********Title page********
10310 GRAPHICS 2+16:COLOR 2
10315 PRINT #6:PRINT #6:PRINT #6:PRINT #6
10320 PRINT #6;"         A"
10325 PRINT #6;"      VACATION"
10330 PRINT #6;"       STORY"
10340 PLOT 1,1
10345 DRAWTO 19,1
10350 DRAWTO 19,9
10355 DRAWTO 1,9
10360 DRAWTO 1,1
10365 RETURN
10600 REM *********Rain**************
10605 FOR LOOP=1 TO 3
10610 GRAPHICS 3+16
10620 FOR COUNTER=1 TO 84
10630 PRINT #6,"+";
10640 NEXT COUNTER
10650 GRAPHICS 0
10660 NEXT LOOP
10670 RETURN
```

```
1900 REM *********Sunshine***********
1902 GRAPHICS 2+16
1910 ? #6
1920 ? #6
1930 ? #6
1950 ? #6
1960 ? #6
1980 ? #6;"THE SUN CAME OUT...."
1990 RETURN
2000 REM *****To be continued******
2010 GRAPHICS 2+16
2020 ? #6
2030 ? #6;"  ....AND HE...."
2040 ? #6
2050 ? #6;" (TO BE CONTINUED)"
2060 ? #6
2065 ? #6
2070 ? #6;"...................."
2080 ? #6;"YOU FINISH THE STORY"
2085 ? #6;"...................."
2095 RETURN
10000 REM *****Going in circles*****
10010 GRAPHICS 7+16:COLOR 2
10020 FOR COUNTER=1 TO 50
10030 Z=Z+0.5
10040 X=SIN(Z)*25:Y=COS(Z)*22
10050 PLOT X+80,Y+45
10060 NEXT COUNTER
10070 RETURN
10100 REM *******Author Page*********
10110 GRAPHICS 2+16
10120 POSITION 4,2:PRINT #6;"*************"
10130 POSITION 4,3:PRINT #6;"*           *"
10135 POSITION 4,4:PRINT #6;"*     BY    *"
10140 POSITION 4,5:PRINT #6;"*           *"
10150 POSITION 4,6:PRINT #6;"* YOUR NAME *"
10155 POSITION 4,7:PRINT #6;"*           *"
10160 POSITION 4,8:PRINT #6;"*************"
10170 RETURN
```

A VACATION STORY

```
20100 REM *******Chirping Birds******
20110 FOR LOOP=1 TO 4
20120 FOR COUNT=1 TO 5
20130 FOR PITCH=1 TO 15
20140 SOUND 2,PITCH,10,8
20150 NEXT PITCH
20160 NEXT COUNT
20170 NEXT LOOP
20180 SOUND 2,0,0,0
20190 RETURN
20300 REM ***Telephone Busy Signal***
20305 FOR RINGS=1 TO 9
20310 SOUND 2,40,6,10
20320 FOR WAIT=1 TO 50:NEXT WAIT
20330 SOUND 2,0,0,0
20340 FOR WAIT=1 TO 25:NEXT WAIT
20350 NEXT RINGS
20360 RETURN
20500 REM *****Steam Locomotive******
20510 FOR LOOP=1 TO 25
20520 FOR LOUD=10 TO 0 STEP -1
20530 SOUND 0,15,0,LOUD
20540 NEXT LOUD
20550 NEXT LOOP
20560 SOUND 0,0,0,0
20570 RETURN
20900 REM ***********Ocean***********
20910 FOR LOOP=1 TO 2
20920 FOR PITCH=0 TO 12
20930 SOUND 0,PITCH,8,6
20940 FOR WAIT=1 TO 5:NEXT WAIT
20950 NEXT PITCH
20960 FOR PITCH=12 TO 0 STEP -1
20970 SOUND 0,PITCH,8,4
20975 FOR WAIT=1 TO 17:NEXT WAIT
20980 NEXT PITCH
20985 NEXT LOOP
20990 SOUND 0,0,0,0
20995 RETURN
29000 REM ********Wait loop**********
29010 FOR WAIT=1 TO 500:NEXT WAIT
29020 RETURN
```

MODULE 6 - CONDITIONALS

OBJECTIVES

Know how to use IF..THEN in programs.

Be familiar with how AND and OR work.

Given a program that uses AND or OR, be able to predict output.

Know how to use strings in comparisons.

MATERIALS REQUIRED

BASIC Cartridge
BASIC Utility Disk
Camper's Personal Diskette

REFERENCES

Inside Atari Basic - pp. 53-56, 59

Your Atari Computer - pp. 90-91

Atari 400/800 Basic Reference Manual - pp. 18-20

CONTENT

Covers numeric variables and strings with conditionals. Includes a brief introduction to the use of joysticks and controllers. A BASIC Cartridge and the camper's Personal Diskette are required for all of the lessons.

Lesson 1 - IF..THEN

Pages 1-4

New Statement
IF..THEN

Materials
BASIC Utility Disk: FORTUNE

Conditionals and branching are introduced using numeric expressions.

IF..THEN

In this lesson, conditionals and branching are introduced using numeric expressions. The following concepts should be covered at some time during the activities. A transparency and student copies of the code used in this lesson and the next are available.

1. An IF..THEN block is used for an either-or situation. The IF statement contains a comparison. The THEN statement has an action dependent on the IF statement.

2. The computer examines the comparison contained in the IF statement. The THEN statement is executed only if the condition in the comparison is true.

3. Numeric expressions are compared using:

=	equal to
>	greater than
<	less than
<>	not equal to
<=	less than or equal to
>=	greater than or equal to

4. The following comparisons can be made.

```
IF 4*7=90 THEN...
IF INT(RND(1)*4)=3 THEN...
IF NUM1<>NUM2 THEN
IF 8<55 THEN
IF X<=Y THEN
```

<u>Be sure the disk drive, printer, and interface (if you have one) are on before you begin this lesson.</u>

<u>Activity #1</u>

This program is a simple illustration of using <, >, and = to make decisions in a program. Type in the program and run it.

```
10 REM Example of a conditional
20 ? "↰": REM * Clears screen *
30 ?:? "Type in a number: ";
40 INPUT NUM
50 IF NUM>10 THEN ? "That's too big."
60 IF NUM<10 THEN ? "That's too small."
70 IF NUM=10 THEN ? "That's the number I had
in mind.":GOTO 90
80 GOTO 30
90 END
```

IF..THEN
(CONTINUED)

Ask campers to identify the line numbers with comparisons. Step through the program using the numbers 1, 100, and 10 to see how IF..THEN works. Be sure to note that in this program the IF..THEN is used to exit a loop under certain conditions.

Activity #2

Run the program on the BASIC Utility Disk called "FORTUNE" by typing RUN"D:FORTUNE". Then LIST the program and:

1. Find the conditionals.

2. Discuss how the computer randomly selects one of the three possible words to finish each of the sentences.

3. The use of IF..THEN in line 230 to control looping in the program.

A transparency of the program is available. The program is listed below.

```
10 REM Fortune teller
20 PRINT "}":REM Clear screen
30 PRINT :PRINT "I will tell you your fortune."
40 PRINT "Let's see...":PRINT
50 NUM=INT(3*RND(0))
60 FOR WAIT=1 TO 1000:NEXT WAIT
70 PRINT "+ + + + + + + + + + + + +"
80 PRINT "You will become very ";
90 IF NUM=0 THEN PRINT "rich."
100 IF NUM=1 THEN PRINT "poor."
110 IF NUM=2 THEN PRINT "powerful."
120 NUM=INT(3*RND(0))
130 FOR WAIT=1 TO 750:NEXT WAIT
140 PRINT "You will also be very ";
150 IF NUM=0 THEN PRINT "happy."
160 IF NUM=1 THEN PRINT "famous."
170 IF NUM=2 THEN PRINT "popular."
180 PRINT "+ + + + + + + + + + + + + +"
190 GOTO 30
```

FORTUNE TELLER

```
10 REM Fortune teller
20 PRINT "}":REM Clear screen
30 PRINT :PRINT "I will tell you your fortune."
40 PRINT "Let's see...":PRINT
50 NUM=INT(3*RND(0))
60 FOR WAIT=1 TO 1000:NEXT WAIT
70 PRINT "+ + + + + + + + + + + + +"
80 PRINT "You will become very ";
90 IF NUM=0 THEN PRINT "rich."
100 IF NUM=1 THEN PRINT "poor."
110 IF NUM=2 THEN PRINT "powerful."
120 NUM=INT(3*RND(0))
130 FOR WAIT=1 TO 750:NEXT WAIT
140 PRINT "You will also be very ";
150 IF NUM=0 THEN PRINT "happy."
160 IF NUM=1 THEN PRINT "famous."
170 IF NUM=2 THEN PRINT "popular."
180 PRINT "+ + + + + + + + + + + + +"
190 GOTO 30
```

IF..THEN
(CONTINUED)

Activity #3

This program might be used at the beginning of a game for two people to settle the age old problem of who gets to go first. Type it in and then run it several times.

```
10 REM **   Coin Toss   **
20 DIM R$(1)
30 ? "↰":REM Clears screen
40 ? "One person chooses heads, the"
50 ? "other person chooses tails.":?
60 ? "Press return when you have decided.";
70 INPUT R$: ?
80 IF INT(2*(RND(1)))<1 THEN ? "The person
who chose heads goes first.":GOTO 100
90 ? "The person who chose tails goes first."
100 GOTO 100
```

Discuss how the conditional in line 80 controls the flow of the program. Explain how line 70 enables one to let the user control the program. That is, the program stops at line 70 until the user presses RETURN. Talk about the use of DIM R$(1) at the beginning of the program.

IF..THEN WITH STRINGS

When campers finish this lesson they should know the following:

1. Strings can be used in IF..THEN blocks.

2. IF..THEN evaluates a comparison and if true, carries out an action.

3. String comparisons use = and <>. They may also use <, >, <=, and >=, but those will not be considered in this lesson.

4. When the comparison is evaluated, the strings must match exactly, for example, "Yes"="Yes" is true, but "Yes"="YES" is not true.

<u>Activity #1</u>

Type in the following program. Ask campers to predict what the output will be before they run the program.

```
10 REM * A matter of taste *
20 REM * Using strings in IF..THEN blocks *
30 DIM ANS$(3)
40 ? "↰":REM * Clears screen *
50 ? "Do you like chocolate? (Type YES or NO)"
60 INPUT ANS$
70 IF ANS$="YES" THEN ? "You have good taste.":
GOTO 90
80 ? "Your taste is questionable!"
90 END
```

Discuss the program. Ask campers to explain lines 60 and 70. Try typing "YUP" and discuss the unfortunate results. Change "YES" in line 60 to "Yes", "yes", "y", and/or "Y" and run the program again to illustrate the fact that the strings must match exactly.

IF..THEN WITH STRINGS
(CONTINUED)

Activity #2

Run the program on the BASIC Utility Disk called PASSWORD by typing RUN"D:PASSWORD". The program is printed below for your information. Have campers (if they do not get the password the first time) LOAD the program, LIST it, and find out why it destroyed itself when an incorrect word was given. Caution campers not to use NEW in their programs! This should be a lesson to them.

PASSWORD PROGRAM

```
10 REM * Self destructive program *
20 ? "}":REM * Clears Screen *
30 DIM WORD$(25),PASSWORD$(25)
40 ? "If you want to read my message,"
50 ? "you must give me the password."
60 ? "If you give me the wrong word,"
70 ? "the program destroys itself."
80 INPUT WORD$
90 PASSWORD$="ATARI"
100 IF WORD$<>PASSWORD$ THEN NEW
110 ? :? "Congratulations!  You guessed"
120 ? "the password."
130 END
```

IF..THEN WITH STRINGS
(CONTINUED)

Activity #3

Run the program on the BASIC Utility Disk called WHOAMI. In it the user uses a clue to identify historical figures who are famous for their contributions to computing. The program uses strings in comparisons. It is available on the page called "WHO AM I?". After campers run the program, they should list the program to the printer (using LIST"P:"). Camper copies are available if you are unable to get a listing from the printer. Examine the program. Be sure to cover at least the following items.

1. Find the main program and each of the subroutines.

2. Discuss the subroutine in lines 2700-2770, including its relationship to each of the "CORRECTANS$=____" comparisons in the main program.

3. Point out the fact that the program would have been much longer without the use of subroutines.

4. Talk about formatting on the screen and how important it can be to the user (eg. What would happen if the possible answers were not available each time a question were asked?)

5. Challenge campers to change the program so that

   a. it gives the user a certain number of chances before giving the answer.

   b. it does not give the answer unless the user asks for it.

   c. it adds more people to the program.

WHO AM I?

```
10 REM *Who Am I-Names in Computing*
20 DIM CORRECTANS$(1),USERANS$(1),R$(1)
30 GOSUB 2600:REM **   Directions **
40 GOSUB 2100:REM **    Babbage   **
50 GOSUB 2800:REM **    Answers   **
60 CORRECTANS$="E"
70 GOSUB 2700:REM ** Answer Input **
80 GOSUB 29000:REM **  Wait Loop  **
90 GOSUB 2500:REM **   Hollerith **
100 GOSUB 2800:REM **  Answers    **
110 CORRECTANS$="B"
120 GOSUB 2700:REM ** Answer Input**
130 GOSUB 29000:REM **  Wait Loop **
140 GOSUB 2200:REM **     Ada     **
150 GOSUB 2800:REM **   Answers   **
160 CORRECTANS$="C"
170 GOSUB 2700:REM ** Answer Input**
180 GOSUB 29000:REM **  Wait Loop **
190 GOSUB 2300:REM **   Pascal    **
200 GOSUB 2800:REM **   Answers   **
210 CORRECTANS$="A"
220 GOSUB 2700:REM ** Answer Input**
230 GOSUB 29000:REM ** Wait Loop  **
240 GOSUB 2400:REM **  Boole      **
250 GOSUB 2800:REM **   Answers   **
260 CORRECTANS$="D"
270 GOSUB 2700:REM ** Answer Input**
280 GOSUB 29000:REM ** Wait Loop  **
290 END
2100 REM ***    Charles Babbage    ***
2110 ? "}":PRINT
2120 ? "An English mathematician and"
2130 ? "inventor who is often called"
2140 ? "the Father of Computing.  He"
2150 ? "said, 'I am thinking that all"
2160 ? "those tables might be"
2170 ? "calculated by machinery.'"
2180 ? :?
2190 RETURN
2200 REM ******       ADA       *****
2210 ? "}":PRINT
2220 ? "An exceptional English mathematician"
2230 ? "who is credited with being the first"
2240 ? "person to make the statement that"
2250 ? "computers can do only what you"
2260 ? "program them to do.  Wrote about"
2270 ? "Babbage's Analytical Engine."
2280 PRINT :PRINT
2290 RETURN
```

WHO AM I?

```
2300 REM ***     Blaise Pascal      ***
2310 ? "}":PRINT
2320 ? "A French mathematician who was"
2330 ? "the first person to invent a"
2340 ? "significant calculating "
2350 ? "machine."
2360 PRINT :PRINT
2370 RETURN
2400 REM ***         Boole          ***
2410 ? "}":PRINT
2420 ? "An English logician.  The"
2430 ? "pioneer of modern symbolic logic."
2440 PRINT :PRINT
2450 RETURN
2500 REM ***     Hollerith        ***
2510 ? "}":?
2520 ? "An American inventor.  The"
2560 ? "first to do a practical"
2570 ? "implementation of punched cards."
2580 PRINT :PRINT
2590 RETURN
2600 REM **  Who Am I - Directions **
2610 ? "}":? :?
2620 ? "          WHO AM I?"
2625 ?
2630 ? "After you read the description"
2640 ? "of the person, choose your"
2650 ? "answer from the names given."
2660 ? "Type in the letter of the"
2670 ? "correct answer.  Each of the"
2680 ? "names is famous in computing."
2685 ? "Press RETURN when you are "
2690 ? "ready to begin."
2695 INPUT R$
2699 RETURN
2700 REM ***  Asks for answer    ***
2710 REM *input with correct answer*
2720 ? :? "Who am I?  Type a letter";
2730 INPUT USERANS$
2740 IF USERANS$=CORRECTANS$ THEN ? "That's correct.":RETURN
2750 ? "That's not my name.  The"
2760 ? "correct answer is ";CORRECTANS$;"."
2770 RETURN
2800 REM ***      Answers         ***
2810 ? :? :?
2820 ? "       A.  Pascal"
2830 ? "       B.  Hollerith"
2840 ? "       C.  Ada"
2850 ? "       D.  Boole"
2860 ? "       E.  Babbage"
2870 RETURN
29000 REM ** Delay Loop  **
29010 FOR WAIT=1 TO 500:NEXT WAIT
29020 RETURN
```

Activity #4

In this activity, campers will examine a number puzzle that combines strings and numeric variables. RUN the program called NUMPUZL and then list it to the printer. A listing is available for the instructor on the pages titled "NUMBER PUZZLE". Camper copies are available if a printer listing is not possible.

Ask campers to use their listing to do the following:

1. Find the beginning and end of the main program. (Lines 10 - 240

2. Explain how the user is able to choose the number puzzle to be solved. (Lines 140 - 180)

3. Describe problems that might arise with string comparisons in the IF..THEN statements. (A solution to the problem of matching upper/lower case and accepting YES or Y or y or yes, etc. is covered in the next lesson.)

4. Decide whether it is possible to write one subroutine that can take the place of each of the subroutines written for the number sequences (as was done in the previous activity in the "Who Am I?" program).

Answers are not given in the program because one of the challenges in Activity #5 is to rewrite the program including the correct answer if the user asks. The first three number sequences are easy. The fourth ("D") is a bit more difficult. The sequence is:

3, 3, 5, 4, 4, 3, 5, 5, ____

and the answer is "4". These numerals stand for the number of letters in each of the words spelling out the numbers one through nine (one, two, three, four, five, etc.).

Activity #5

Challenge campers to change the Number Puzzle program in one of the following ways:

1. Add more sequences.
2. Add hints each time an incorrect answer is input.
3. Give more chances to solve the puzzle.
4. Allow the user to "give up" and get the answer from the computer.
5. Use only one subroutine in place of the four that were used for the sequences. Use the program in the previous activity as a model.
6. Print the answer after a certain number of guesses.

NUMBER PUZZLE

```
10 REM ***  Number Pattern Puzzle ***
20 DIM ANS$(1),MORE$(1)
30 ? "}":? :?
40 ? "This is a number puzzle.  Try"
50 ? "to figure out what the next"
60 ? "number will be in each sequence"
70 ? "These are the sequences:":?
80 ? "A.  1, 3, 5, 7, 11, 13, 17, ___"
90 ? "B.  40, 51, 62, 73, 84, 95, ___"
100 ? "C.  1, 1, 2, 3, 5, 8, 13, 21, ___"
110 ? "D.  3, 3, 5, 4, 4, 3, 5, 5, ___":?
120 ? "What sequence would you like"
130 ? "to try? (Type a letter.)";
140 INPUT ANS$
150 IF ANS$="A" THEN GOSUB 5000
160 IF ANS$="B" THEN GOSUB 5100
170 IF ANS$="C" THEN GOSUB 5200
180 IF ANS$="D" THEN GOSUB 5300
200 ? :? "Do you want to try another"
210 ? "sequence? (Type Y or N)";
220 INPUT MORE$
230 IF MORE$="Y" THEN ? "}":GOTO 70
240 END
5000 REM ****prime number sequence***
5005 REM ****                     ***
5010 ? "}":? :? :REM * clears screen  *
5020 ? "Type a number to finish this"
5025 ? "sequence. You get three chances.":?
5030 ? "1, 3, 5, 7, 11, 13, 17, ____"
5040 FOR COUNT=1 TO 3
5050 ? :? "NUMBER";
5060 INPUT NUM
5070 IF NUM=19 THEN ? "That's correct.":RETURN
5080 ? "It's not ";NUM;".  These are prime numbers."
5090 NEXT COUNT
5095 RETURN
```

```
5100 REM ****add eleven sequence*****
5105 REM ****                   *****
5110 ? "}":? :? :REM * clears screen  *
5120 ? "Type a number to finish this"
5125 ? "sequence. You get three chances.":?
5130 ? "40, 51, 62, 73, 84, 95, ____"
5140 FOR COUNT=1 TO 3
5150 ? :? "NUMBER";
5160 INPUT NUM
5170 IF NUM=106 THEN ? "That's correct.":RETURN
5180 ? "It's not ";NUM;"."
5190 NEXT COUNT
5195 RETURN
5200 REM ****Fibonacci sequence******
5205 REM ****                 ******
5210 ? "}":? :? :REM * clears screen  *
5220 ? "Type a number to finish this"
5225 ? "sequence. You get three chances.":?
5230 ? "1, 1, 2, 3, 5, 8, 13, 21, ____"
5240 FOR COUNT=1 TO 3
5250 ? :? "NUMBER";
5260 INPUT NUM
5270 IF NUM=34 THEN ? "That's correct.":RETURN
5280 ? "It'S not ";NUM;".  It's a Fibonacci."
5290 NEXT COUNT
5295 RETURN
5300 REM **number of letters in nums*
5305 REM ***                     ***
5310 ? "}":? :? :REM * clears screen  *
5320 ? "Type a number to finish this"
5325 ? "sequence. You get three chances.":?
5330 ? "3, 3, 5, 4, 4, 3, 5, 5, ____"
5340 FOR COUNT=1 TO 3
5350 ? :? "NUMBER";
5360 INPUT NUM
5370 IF NUM=4 THEN ? "That's correct.":RETURN
5380 ? "It's not ";NUM;"."
5390 NEXT COUNT
5395 RETURN
```

IF..THEN
CAMPER COPY

```
10 REM Example of a conditional
20 ? "ʃ": REM * Clears screen *
30 ?:? "Type in a number: ";
40 INPUT NUM
50 IF NUM>10 THEN ? "That's too big."
60 IF NUM<10 THEN ? "That's too small."
70 IF NUM=10 THEN ? "That's the number I had
in mind.":GOTO 90
80 GOTO 30
90 END
```

```
10 REM **   Coin Toss   **
20 DIM R$(1)
30 ? "ʃ":REM Clears screen
40 ? "One person chooses heads, the"
50 ? "other person chooses tails.":?
60 ? "Press return when you have decided.";
70 INPUT R$: ?
80 IF INT(2*(RND(1)))<1 THEN ? "The person
who chose heads goes first.":GOTO 100
90 ? "The person who chose tails goes first."
100 GOTO 100
```

```
10 REM * A matter of taste *
20 REM * Using strings in IF..THEN blocks *
30 DIM ANS$(3)
40 ? "ʃ":REM * Clears screen *
50 ? "Do you like chocolate? (Type YES or NO)"
60 INPUT ANS$
70 IF ANS$="YES" THEN ? "You have good taste.":
GOTO 90
80 ? "Your taste is questionable!"
90 END
```

OR / AND

The use of AND and OR is confusing to some people. The activities that follow are the minimum that should be done to insure understanding of these logic operators. When given examples of AND and OR used in programs, campers should be able to explain how they work. Campers should also know when it is appropriate to use AND and OR. A transparency and camper copies of the code to be typed in is available.

Activity #1

This simple program illustrates the use of OR. Type it in and run it, trying different colors as input.

```
 10 REM * Use of OR and AND *
 20 ? "↰":REM Clear screen
 30 DIM COLOR$(15)
 40 ? "Type in your favorite color.  Use"
 50 ? "all capital letters like this, RED.":?
 60 INPUT COLOR$
 70 IF COLOR$="RED" OR COLOR$="YELLOW" THEN
 ? "That is one of my favorites, too!":GOTO 90
 80 ?  "Your taste in colors is different
 than mine."
 90 END
```

Change the OR in line 70 to AND. Run the program again and try RED and then YELLOW when the program asks for input. Finally, change the following lines in the program:

```
 30 DIM COLOR1$(15), COLOR2$(15)
 40 ? "Type in two colors.  Use"
 60 ? "What is the first color";
 70 INPUT COLOR1$
 80 ? "What is the second color";
 90 INPUT COLOR2$
100 IF COLOR1$="RED" AND COLOR2$="YELLOW" THEN
 ? "Those are my favorite colors.":GOTO 120
110 ? "I'm not fond of those colors."
120 END
```

Run it and try different color combinations. Discuss the effect of AND in the program.

Activity #2

A common use of AND is to control looping by establishing a condition that will make the program exit the loop. Have campers run the program on the BASIC Utility Disk called "USINGAND". The program is printed below and on the page titled "USING AND". Campers should list the program to the printer. If that is not possible, use the "USING AND" page.

1. Ask campers to find the answer to the question in the program (line 110).

2. Run the program again, typing in the correct answer. Observe what happens to the values of "COUNT" and "CORRECT" when the correct answer is input.

3. Try typing in the correct answer on the second chance and the third chance to see how the values of the variables used in the IF..AND..THEN change.

4. Ask campers to find the line that controls the loop that repeats inputing answers (line 150).

5. Discuss initialization of variables in line 30.

6. A counter has not been discussed prior to this program. Talk about how COUNT is used in line 120.

```
10 REM ***   Using AND       ***
20 DIM ANS$(10),CORRECT$(1)
30 CORRECT$="N":COUNT=0
40 ? "}":?
50 ? "Who published 'On Computable"
60 ? "Numbers', one of the most important"
70 ? "papers in the foundations of"
80 ? "computer science?  You get three"
90 ? "chances to give the right answer.":?
100 INPUT ANS$
110 IF ANS$="Turing" THEN CORRECT$="Y"
120 COUNT=COUNT+1
130 ? "COUNT = ";COUNT
140 ? "CORRECT = ";CORRECT$:?
150 IF COUNT<>3 AND CORRECT$="N" THEN ? "Try again: ";:GOTO 100
160 IF CORRECT$="Y" THEN ? "Yes, it was Alan Matheson Turing.":GOTO 200
170 ? "This was a difficult question.  The"
180 ? "answer can be found on page 79 of"
190 ? "the book, 'The Making of the Micro'."
200 END
```

USING AND

```
10 REM ***   Using AND        ***
20 DIM ANS$(10),CORRECT$(1)
30 CORRECT$="N":COUNT=0
40 ? "}":?
50 ? "Who published 'On Computable"
60 ? "Numbers', one of the most important"
70 ? "papers in the foundations of"
80 ? "computer science?  You get three"
90 ? "chances to give the right answer.":?
100 INPUT ANS$
110 IF ANS$="Turing" THEN CORRECT$="Y"
120 COUNT=COUNT+1
130 ? "COUNT = ";COUNT
140 ? "CORRECT = ";CORRECT$:?
150 IF COUNT<>3 AND CORRECT$="N" THEN ? "Try again: ";:GOTO 100
160 IF CORRECT$="Y" THEN ? "Yes, it was Alan Matheson Turing.":GOTO 200
170 ? "This was a difficult question.  The"
180 ? "answer can be found on page 79 of"
190 ? "the book, 'The Making of the Micro'."
200 END
```

OR / AND
(CONTINUED)

Activity #3

This example shows campers how to use OR to help with the problem of matching strings exactly when asking a user for input in a program. Type it in and run it.

```
10 REM Use of OR to solve one input problem
20 ? "↰":REM Clears screen
30 DIM ANS$(3)
40 ? "Is your favorite car a Porsche";
50 INPUT ANS$
60 IF ANS$="YES" OR ANS$="Y" OR ANS$="yes"
OR ANS$="y" OR ANS$="Yes" THEN ? "What class!":
GOTO 80
70 ? "It isn't.  I'm amazed!"
80 END
```

Remind campers that they cannot use ANS$="YES" or "Y" or "yes". This is a typical error made by beginning programmers. ANS$ must be used with each comparison as illustrated in line 60.

OR / AND
CAMPER COPY

```
10 REM * Use of OR and AND *
20 ? "↰":REM Clear screen
30 DIM COLOR$(15)
40 ? "Type in your favorite color.  Use"
50 ? "all capital letters like this, RED.":?
60 INPUT COLOR$
70 IF COLOR$="RED" OR COLOR$="YELLOW" THEN
? "That is one of my favorites, too!":GOTO 90
80 ?  "Your taste in colors is different
than mine."
90 END
```

```
 30 DIM COLOR1$(15), COLOR2$(15)
 40 ? "Type in two colors.  Use"
 60 ? "What is the first color";
 70 INPUT COLOR1$
 80 ? "What is the second color";
 90 INPUT COLOR2$
100 IF COLOR1$="RED" AND COLOR2$="YELLOW" THEN
 ? "Those are my favorite colors.":GOTO 120
110 ? "I'm not fond of those colors."
120 END
```

```
10 REM Use of OR to solve one input problem
20 ? "↰":REM Clears screen
30 DIM ANS$(3)
40 ? "Is your favorite car a Forsche";
50 INPUT ANS$
60 IF ANS$="YES" OR ANS$="Y" OR ANS$="yes"
OR ANS$="y" OR ANS$="Yes" THEN ? "What class!":
GOTO 80
70 ? "It isn't.  I'm amazed!"
80 END
```

USING A JOYSTICK

The purpose of this lesson and the next one is to introduce the use of joysticks and paddles. It is included in this module because knowledge of conditionals can be applied in a way that is fun and interesting, without requiring lengthy programming assignments. Discuss the following before you begin the activities in this lesson.

1. The STICK function reads the joystick in ATARI BASIC. It takes the form STICK(0), where 0 is the port used by the joystick. There are four ports on the Atari 800. The numbers used by the STICK function are 0, 1, 2, and 3.

2. The computer translates the joystick positions as certain numbers. When you want to program the computer to use the joystick, you use IF..THEN to do something if a particular number is found.

3. The position numbers are shown below.

```
                 forward
                   14

       up,left             up,right
             10          6

                 resting
   left 11         15          7 right


              9          5
      down,left             down,right

                   13
                  back
```

4. The STRIG function reads the joystick trigger button. It returns a value of 0 if the trigger is being pressed, 1 if it is not.

<u>In this lesson, you will need a joystick for each computer.</u>

A transparency and worksheets are available for the code used in the lesson.

USING A JOYSTICK
(CONTINUED)

Activity #1

Type in and run the following to see how the joystick values are read.

```
10 REM * Practice with the joystick *
20 NUM=STICK(0)
30 IF NUM=15 THEN ? "RESTING"
40 IF NUM=14 THEN ? "GOING FORWARD"
50 IF NUM=13 THEN ? "GOING BACKWARD"
60 IF NUM=11 THEN ? "LEFT"
70 IF NUM=7 THEN ? "RIGHT"
80 IF NUM=10 THEN ? "UP & LEFT"
90 IF NUM=6 THEN ? "UP & RIGHT"
100 IF NUM=9 THEN ? "DOWN & LEFT"
110 IF NUM=5 THEN ? "DOWN & RIGHT"
120 ? "Joystick = ";NUM:?
130 FOR WAIT=1 TO 250:NEXT WAIT
140 GOTO 20
```

Activity #2

Use this short program to see the value change when you press and release the joystick button. STRIG stands for STICK TRIGger. STRIG returns the value of 1 until the button is pressed, and then the value returned is zero.

```
10 NUM=STRIG(0)
20 PRINT NUM
30 GOTO 10
```

Activity #3

Try this program for random sound using the joystick.

```
10 REM *  Joystick and SOUND
20 PITCH=STICK(0)
30 SOUND 0,PITCH*10,10,10
40 GOTO 10
```

USING A JOYSTICK

Activity #4

Run the program on the BASIC Utility Disk called "JSTICK". The copy of the program below is for the instructor's use. Campers should list the program to the printer for their copy. Be sure that campers press the joystick button to see the square change colors. At some point, they will get a cursor out of range error. Be sure to discuss why that happens when you discuss the program.

```
10 REM * Experimenting with a Joystick *
15 GRAPHICS 3+16
20 X=39:Y=19:C=1
30 XOLD=X:YOLD=Y
40 STK=STICK(0):IF STK<>15 THEN GOSUB 200
50 COLOR 0:PLOT XOLD,YOLD
60 COLOR C:PLOT X,Y
70 IF STRIG(0)=0 THEN GOSUB 300
80 GOTO 30
200 IF (STK=5) OR (STK=6) OR (STK=7) THEN X=X+1
210 IF (STK=9) OR (STK=10) OR STK=(11) THEN X=X-1
220 IF (STK=10) OR (STK=6) OR (STK=14) THEN Y=Y-1
230 IF (STK=9) OR (STK=13) OR (STK=5) THEN Y=Y+1
240 RETURN
300 C=C+1
310 IF C=4 THEN C=1
320 RETURN
```

After campers have run the program and listed it, step through each line and ask them to explain what is happening.

Activity #5

Just for fun, RUN the program on the BASIC Utility Disk called "JOYARROW". It demonstrates use of the joystick and controllers to control an animated arrow. The program is available on the page titled "JOYARROW".

JOYARROW

```
10 REM CONTROLLED ARROW
20 GRAPHICS 7
30 PRINT "JOYSTCK IN PORT 1: CONTROLS DIRECTION.";
40 PRINT "              TRIGGER BUTTON CHANGES HUE.";
50 PRINT "PADDLE IN PORT 2:  CONTROLS LUMINANCE.";
60 X=15:Y=40:HUE=0:LUM=14
70 SETCOLOR 0,HUE,LUM
80 COLOR 1:GOSUB 300
90 COLOR 0:GOSUB 300
100 IF STRIG(0)=0 THEN HUE=HUE+1
110 IF HUE>15 THEN HUE=0
120 LUM=INT(PADDLE(2)/16)
130 IF STICK(0)=14 THEN Y=Y-1
140 IF STICK(0)=6 THEN X=X+1:Y=Y-1
150 IF STICK(0)=7 THEN X=X+1
160 IF STICK(0)=5 THEN X=X+1:Y=Y+1
170 IF STICK(0)=13 THEN Y=Y+1
180 IF STICK(0)=9 THEN X=X-1:Y=Y+1
190 IF STICK(0)=11 THEN X=X-1
200 IF STICK(0)=10 THEN X=X-1:Y=Y-1
210 GOTO 70
220 STOP
300 PLOT X,Y
310 DRAWTO X-15,Y
320 DRAWTO X-15,Y+2
330 DRAWTO X,Y+2
340 PLOT X+2,Y+1
350 DRAWTO X-6,Y-3
360 PLOT X+2,Y+1
370 DRAWTO X-6,Y+5
380 RETURN
390 END
400 REM *
410 REM *
420 REM *
430 REM *
440 REM *
450 REM *         JOYSTICK                      PADDLE
460 REM *
470 REM *
480 REM *            14
490 REM *
500 REM *        10        6
510 REM *
520 REM *    11      15       7          228            1
530 REM *
540 REM *         9        5
550 REM *
560 REM *            13
```

PADDLE CONTROLLERS

This is an introduction to the use of paddles. Programming with paddles is similar to that of joysticks, except that the computer interprets positions from 1 to 228. Atari BASIC reads the paddle with the PADDLE function. The paddle button uses PTRIG and works the same way the joystick function works. The activities in this lesson are intended to allow the camper to experiment with the paddle functions.

<u>You will need paddles for each computer in order to do this lesson.</u>

Activity #1

Type in the following short program and run it.

```
10 REM * Using a paddle *
20 PITCH=PADDLE(0)
30 PRINT PITCH
40 SOUND 0,PITCH,10,10
50 GOTO 10
```

Ask campers to describe the relationship of the numeric values that appear on the screen to the pitch of the sound they hear.

Activity #2

Now try this program. When it is run, the button should be pushed to control the volume of the sound.

```
10 REM *Experimenting with paddles and sound.*
20 VOL=10
30 SOUND 0,PADDLE(0),10,VOL
40 IF PTRIG(0)=0 THEN GOSUB 100
50 FOR DELAY=1 TO 40:NEXT DELAY
60 GOTO 30
100 VOL=VOL+1
110 IF VOL=16 THEN VOL=0
120 RETURN
```

Step through the program with the campers and then add the following lines to further illustrate what is happening.

```
25 PRINT PADDLE(0)
105 PRINT ,VOL
```

Run the program to see the change in numeric values as the paddle is changed.

Activity #3

Challenge campers to change the programs in the first two activities, so that they can use two paddles at once.

JOYSTICKS AND CONTROLLERS
CAMPER COPY

```
10 NUM=STRIG(0)
20 PRINT NUM
30 GOTO 10
```

```
10 REM *  Joystick and SOUND
20 PITCH=STICK(0)
30 SOUND 0,PITCH*10,10,10
40 GOTO 10
```

```
10 REM * Using a paddle *
20 PITCH=PADDLE(0)
30 PRINT PITCH
40 SOUND 0,PITCH,10,10
50 GOTO 10
```

```
10 REM *Experimenting with paddles and sound.*
20 VOL=10
30 SOUND 0,PADDLE(0),10,VOL
40 IF PTRIG(0)=0 THEN GOSUB 100
50 FOR DELAY=1 TO 40:NEXT DELAY
60 GOTO 30
100 VOL=VOL+1
110 IF VOL=16 THEN VOL=0
120 RETURN
```

Add these lines:

```
25 PRINT PADDLE(0)
105 PRINT ,VOL
```

JOYSTICKS AND CONTROLLERS
CAMPER COPY

```
10 NUM=STRIG(0)
20 PRINT NUM
30 GOTO 10
```

```
10 REM *  Joystick and SOUND
20 PITCH=STICK(0)
30 SOUND 0,PITCH*10,10,10
40 GOTO 10
```

```
10 REM * Using a paddle *
20 PITCH=PADDLE(0)
30 PRINT PITCH
40 SOUND 0,PITCH,10,10
50 GOTO 10
```

```
10 REM *Experimenting with paddles and sound.*
20 VOL=10
30 SOUND 0,PADDLE(0),10,VOL
40 IF PTRIG(0)=0 THEN GOSUB 100
50 FOR DELAY=1 TO 40:NEXT DELAY
60 GOTO 30
100 VOL=VOL+1
110 IF VOL=16 THEN VOL=0
120 RETURN
```

Add these lines:

```
25 PRINT PADDLE(0)
105 PRINT ,VOL
```

MODULE #7 - INDIVIDUAL PROJECT

OBJECTIVES

Be able to define "Utility" and state purposes of utility programs in general.

Apply program planning techniques learned in Module 5

Learn to use the following utilities:
PAINT
PLAYER MAKER
ANIMATE

Produce a "computer show" that tells a story and uses a painting, players, and animation.

MATERIALS REQUIRED

BASIC Cartridge
Camper's Personal Diskette
GENERAL UTILITY DISKETTE
PAINT
PLAYER MAKER
"Creating a Computer Show" Documentation
"ANIMATION" Documentation

REFERENCES

You should read all of the documentation accompanying the utility packages before you teach the lessons in this module.

CONTENT

<u>Lesson 1 - Individual Project Introduction</u>

Pages 1-2

An introduction for the instructor. Read carefully before you begin teaching.

Lesson 2 - Creating a Player

Page 3

Materials
- PLAYER MAKER
- Camper's Personal Diskette

PLAYER MAKER is used to create a player (or character) for the story. The player will be used in Lesson 4.

Lesson 3 - Using the GENERAL UTILITY DISKETTE

Pages 4-5

Materials
- GENERAL UTILITY DISKETTE
- Camper's Personal Diskette

Campers learn how to use the programs on the GENERAL UTILITY DISKETTE to help with the production of their "computer show."

Lesson 4 - Animate

Page 6

Materials
- GENERAL UTILITY DISKETTE: ANIMATE
- Camper's Personal Diskette
- Joystick for every computer

Animation is created for each player. The programming is done to complete the show.

Lesson 5 - Individual Project

Pages 7-8

Campers use what they have learned in the previous lessons to produce their own show.

INDIVIDUAL PROJECT INTRODUCTION

In this module, campers will learn to produce a "computer show" using PAINT, PLAYER MAKER, MUSIC PLAYER (stored as AUTORUN.SYS on the GENERAL UTILITY DISKETTE), and ANIMATE. The first program they do will allow few choices as to what is included. It is designed to serve as a model for the program they produce on their own in the last lesson of the module. The model will include a picture, figures that move on the picture, and music that plays in the background. To successfully complete the activities, campers must have some knowledge of BASIC, and they must know how to use the Atari computer system.

Each lesson covers a different part of the process used to combine the utilities. If the lessons are done in order and proper instruction is given, campers should be able to use the utilities to create their own show with very little assistance from an instructor. Thus, the individual camper projects in the fourth lesson could be used to provide activities for campers who do not need to be part of a lesson and have nothing structured to do while the rest of the group is receiving instruction on other parts of the BASIC curriculum. Because the projects are very time consuming, campers should be warned that they may have to do some work outside of class in order to finish.

It will probably be the case that two campers will have to work together in doing the model program. Use one Personal Diskette to store the program. If the second camper wants a copy of the finished product on his or her diskette, a TA or the teacher should copy the necessary files.

In order to make the module more than merely a "canned" activity, you should cover the following material at some time during the lessons.

1. Discuss the definition and purposes of utility packages in general. As you use each of the utilities, talk about more specific uses for each one.

2. Use this opportunity to talk about the importance of planning a computer program before you begin to work on the computer. Encourage campers to outline their final project before they begin to create the picture, players, etc.

3. If the Subroutines module has been completed, discuss the use of subroutines in it.

4. Introduce (or review):

   ```
   SAVE"D:___________"
   LOAD"D:___________"
   LIST"D:___________"
   ENTER"D:__________"
   ```

   Emphasize the need for using descriptive file names when storing programs or subroutines. Talk about legal names and what happens if you accidentally use a name that has already been used to store a program. (The first program is overwritten.)

5. Explain what is happening when DOS is used in the procedure. For example, do more than just have campers type in "DOS" and then "O". Talk about what happens when each is typed. Campers should also know how to look at the directory on the GENERAL UTILITY DISKETTE in order to use paintings and music other than the ones they use in the directed lessons.

6. It will be important that you make very clear which parts of the program can be created by the campers and which must be followed exactly as they are written. A single deviation from the directions for putting the various parts of the program together could have disastrous effects.

Be sure you have read the documentation that accompanies each utility package before you teach the lesson using the utility.

CREATING A PLAYER

Campers will use PLAYER MAKER to create a player or character for their story. If you have not read the PLAYER MAKER documentation, do so before you begin the lesson. The diskette and documentation are stored in sets of six in the library.

I. Do a demonstration of how to make a player using PLAYER MAKER. Show how to use the menu, move around the system, and create a player. Complete a player and save it on a diskette. The player will be used in the animation lesson. Talk about how to make pictures using squares. Some campers may need to plan their players before they make them.

II. Instruct campers step by step in making a player. If there are two people at a computer, the show that is created will be cooperatively produced. One person should do Player 0 and the other person should do Player 1. Be sure that the BASIC cartridge is in place. Tell campers to:

1. Insert the PLAYER MAKER diskette and load the PLAYER MAKER program. Once the program is loaded, remove the diskette and insert one camper's Personal Diskette into the disk drive, so that the objects created can be saved.

2. Design the players. Some campers may need suggestions for figures to design. It is important that the figures remain simple. If they do not, the activity will take too much time. Some letters of the alphabet are easy. A smiling or frowning face has appeal and is not too difficult. Figures with rounded edges are more difficult to make and may take too much time for this activity.

3. Use the following procedure to save the players.

   a. Type "F".
   b. Move the cursor to SAVE PLAYERS AND END PROGRAM.
   c. Press the joystick button or the space bar.
   d. Type "S" to save a subroutine.
   e. Use the filename, "PLAYER". Caution campers that if "PLAYER" is not used, they will have trouble following directions later on.

USING THE GENERAL UTILITY DISKETTE

The steps for copying files from the GENERAL UTILITY DISKETTE are listed below. If campers are not familiar with using DOS, this would be an appropriate time to show how to use the "DISK DIRECTORY", "RUN CARTIRDGE", AND "DUPLICATE FILE" items. You might also want to show how to list the directory to the printer using the disk directory. Note that the subroutines that are pictures have the extension ".PIC" and the music subroutines have the extension ".MUS". Have campers do one step at a time as you instruct them. Be sure that the <u>letter</u> "O" is typed when "O" is given in the directions. Caution campers that it is very important that the right disk be in the disk drive when a procedure is being done.

1. Insert the GENERAL UTILITY DISKETTE into the disk drive and type:

```
DOS           (RETURN)
O             (RETURN)
WINTER.PIC    (RETURN)
(RETURN)
```

Tell campers that they are now copying a file containing a picture that was done using PAINT. When they do their own show, they may choose another picture from the GENERAL UTILITY DISKETTE or they may create their own picture using PAINT. In either case, instead of typing WINTER.PIC, they will substitute the name of the picture they want to use in the program in this position. There are more steps needed to use a camper created painting. See the CREATING A COMPUTER SHOW documentation (the page called "FOR THE MORE EXPERIENCED") for instructions.

2. Insert the Personal Diskette into the drive and press (RETURN).

3. Insert the GENERAL UTILITY DISKETTE and type:

```
O             (RETURN)
FUGUE.MUS     (RETURN)
(RETURN)
```

There are other music files available on the GENERAL UTILITY DISKETTE. If campers want to use a different song, they should choose from one of these files. Writing songs is extremely time consuming and is not an appropriate activity at this time.

4. Insert the Personal Diskette and press (RETURN). This step copies the music file on the camper's diskette.

5. Insert the GENERAL UTILITY DISKETTE and type:

```
O            (RETURN)
AUTORUN.SYS (RETURN)
(RETURN)
```

6. Insert the Personal Diskette and press (RETURN). This copies the subroutine that enables the music to play.

7. Insert the GENERAL UTILITY DISKETTE, type B, and press RETURN to get back to BASIC.

8. Be sure the GENERAL UTILITY DISKETTE is in the disk drive and "READY" is on the screen. Type:

```
NEW
ENTER "D:ANIMATE"    (RETURN)
ENTER "D:PAINT"      (RETURN)
ENTER "D:MUSIC"      (RETURN)
```

This step enters the subroutines necessary to use the player animation, painting, and music.

9. Insert the Personal Diskette and type the following <u>EXACTLY AS IT IS WRITTEN:</u>

```
ENTER "D:PLAYER"
20 FILE$="WINTER.PIC":GOSUB PAINT
30 GOSUB TEACH
50 END
SAVE "D:SHOW"
```

This enters the subroutine created when PLAYER MAKER was used in the first lesson. That subroutine is used in the subroutine, "TEACH" in line 30. The painting is displayed by the subroutine in line 20. If another painting is used, the filename on this line should be changed. Everything must be saved at this point.

10. Tell campers that they are now ready to teach animation to the players that were created in the lesson called "MAKING A PLAYER". That activity will be done in the next lesson.

ANIMATE
TEACHING PLAYERS TO MOVE

"ANIMATE" is a BASIC subroutine that is stored on the GENERAL UTILITY DISKETTE. If you have not read the Animation Subroutine Documentation, you should do so before beginning this lesson. Campers must have created a figure using PLAYER MAKER in order to do the activities.

A joystick is required for each computer.

1. Do a demonstration of how to use TEACH and PRACTICE. Be sure that campers understand that TEACH determines the path and PRACTICE allows you to set the speed of movement. They should also be aware that they cannot change the path of a player after it has been "taught", but they can change the timing of previously taught players using PRACTICE.

2. After you have done the demonstration, have campers do their animation. Each camper should teach the player he or she created. The program called "SHOW" must be run in order to begin. Type RUN "D:SHOW".

3. When campers have finished teaching the players, they should complete the following steps:

```
Press  (SYSTEM RESET)
LOAD "D:SHOW"
10 FILE$="FUGUE.MUS":GOSUB MUSIC
30 GOSUB PLAY
40 GOSUB ANIMATE
SAVE "D:SHOW"
```

4. To see the completed show, campers <u>must first turn the computer off</u> and then turn it back on and type RUN "D:SHOW".

5. Be sure campers have saved their program before they turn their computer off.

INDIVIDUAL PROJECT

Campers should have learned how to do a computer show in the first lessons of the module. This lesson gives teaching tips for helping campers to create a show of their own.

1. Encourage planning before the projects are begun. A suggested sequences is to select or create a picture, choose the music to play in the background, and then create the player to fit the setting and the music.

2. Campers may choose to use the subroutines available on the GENERAL UTILITY DISKETTE to provide the picture and music for their show. They should use the DOS directory to see what is on the diskette. The pictures have the extension, .PIC and the music has the extension, .MUS.

2. If campers want to see the pictures or hear the music before they decide what to use in their programs, they should follow to instructions below. Be sure the GENERAL UTILITY DISKETTE is in the disk drive.

TO SEE A PICTURE TYPE:

```
ENTER "D:PAINT"

10 FILE$="__________"
20 GOSUB PAINT
30 GOTO 30
RUN
```

Put the file name of the picture to be viewed in the blank in line 10.

TO LISTEN TO MUSIC:

First turn the computer off and back on. Then type:

```
ENTER "D:MUSIC"
10 FILE$="__________":GOSUB MUSIC
20 GOSUB PLAY
30 GOTO 30
RUN
```

Put the file name of the music to be played in the blank in line 10.

INDIVIDUAL PROJECT
(CONTINUED)

3. If campers do a painting of their own, they will need to complete steps 1 and 2 of the directions on the page called "FOR THE MORE EXPERIENCED" in the "CREATING A COMPUTER SHOW" document.

4. Projects that use more than one painting or several animation sequences will need closer supervision than ones that follow the model done in the first three lessons. Be sure that campers are aware that there can be complications when they begin to add things to their programs. It would be best if they checked with a teacher or a TA before beginning any project they have planned.

5. Remind campers that it takes quite a bit of time to complete a project. If they start close to the end of their stay at camp, they should be careful to plan a show that they are able to complete in the time they have left.

CREATING A COMPUTER SHOW

The following materials are needed for you to create a dramatic computer show.

To create Pictures:

PAINT software (also, PAINT data diskette for saving pictures).
General Utility Diskette: PAINT.CPY, PAINT

To create Music:

Music Composer (or diskette with music data files).
General Utility Diskette: MUSIC, AUTORUN.SYS (from MUSIC PLAYER)

To create Animation:

PLAYER MAKER
General Utility Diskette: ANIMATE

Other:

SCREEN MAKER

Personal Data Diskette -- This is your own diskette which will be used to accumulate all the necessary data and programs.

CREATING A COMPUTER SHOW

The following instructions provide a good starting point for you to learn to create a computer show. Many of the options available to you can be explored later, after you have had a chance to get acquainted with the more straightforward procedure below.

1. The first step is to create some figures or objects that you will later move around (animate) on top of a painting. Insert a BASIC cartridge and load the PLAYER MAKER program. Once the program is loaded, remove the diskette and insert your own Personal Diskette into the disk drive in order to save the objects you created using PLAYER MAKER. Refer to the PLAYER MAKER documentation pages 1 through 6 only. When you finish designing your players, (1) type "F", (2) move the cursor to SAVE PLAYERS AND END PROGRAM, (3) press the joystick button, (4) type "S" to save a subroutine, and (5) provide the filename PLAYER.

2. Now you must copy some files from the GENERAL UTILITY DISKETTE to your own Personal Diskette. Insert the GENERAL UTILITY DISKETTE into the disk drive and type the following:

```
DOS        (RETURN)
Press the letter "O" and (RETURN)
WINTER.PIC     (RETURN)
(RETURN)
```

   Now insert your own Personal Diskette into the drive and press (RETURN).
   Then switch diskettes again so that the GENERAL UTILITY DISKETTE is in the disk drive.

```
Press the letter "O" and (RETURN)
Type:  FUGUE.MUS     (RETURN)
(RETURN)
```

   Insert your Personal Diskette and press (RETURN).
   Switch back to the GENERAL UTILITY DISKETTE.

```
Press the letter "O" and (RETURN)
Type:  AUTORUN.SYS     (RETURN)
(RETURN)
```

   Insert your Personal Diskette and press (RETURN).
   Switch back to the GENERAL UTILITY DISKETTE.
   Press the letter "B" and (RETURN) to get back to BASIC.

3. Now you must enter some programs into computer memory. Type the following (with the GENERAL UTILITY DISKETTE in the disk drive and "READY" on your screen):

```
NEW
ENTER "D:ANIMATE"      (RETURN - wait for disk drive to stop)
ENTER "D:PAINT"        (RETURN - wait for disk drive to stop)
ENTER "D:MUSIC"        (RETURN - wait for disk drive to stop)
```

Now insert your own Personal Diskette into the disk drive and type the following:

```
ENTER "D:PLAYER"     (RETURN - wait for disk drive to stop)
20 FILE$="WINTER.PIC":GOSUB PAINT     (RETURN)
40 GOSUB TEACH     (RETURN)
50 END     (RETURN)
SAVE "D:SHOW"     (RETURN)
```

4. Now you are ready to teach animation to the players you created. See the "Animation Subroutine Documentation" and when ready, type: RUN. When you finish teaching your players, go on to step 5.

5. Now you can complete your entire computer show. Do the following:

```
Press (SYSTEM RESET).
LOAD "D:SHOW"     (RETURN)
10 FILE$="FUGUE.MUS":GOSUB MUSIC     (RETURN)
30 GOSUB PLAY     (RETURN)
40 GOSUB ANIMATE     (RETURN)
SAVE "D:SHOW"     (RETURN)
```

6. To see your completed show, first turn the computer off and then turn it back on. Then type:

```
RUN "D:SHOW"     (RETURN)
```

FOR THE MORE EXPERIENCED

1. Create one or more paintings using the PAINT software. Save your paintings to files on a PAINT Data Diskette -- you will probably find such a diskette in the vicinity of the PAINT program.

2. Load and run the program called PAINT.CPY from the General Utility Diskette. This will copy the paintings you created from the PAINT Data Diskette to your own Personal Diskette.

3. Use the PLAYER MAKER program to create figures that you will later move about on your pictures. Save the "player subroutines" you create to your Personal Diskette. Use the filename PLAYER.

4. If you are a little more experienced, and you want to have text on your screen combined with your paintings, then use SCREEN MAKER to design a screen layout. Use GRAPHICS 7 as the background mode and then put text modes (GRAPHICS 0, 1, or 2) where you want them. Be sure you leave at least one scan line of GRAPHICS 7 at the very top of your screen. Save your screen subroutine to your Personal Diskette in a file called SCREEN.

5. Use the MUSIC COMPOSER cartridge to create the music you would like in your show and save this on your Personal Diskette. Or, if you like, copy music routines using DOS from the General Utility Diskette to your Personal Diskette.

Your Personal Diskette should now include the following files:

PAINT data files:	filename.PIC
Player subroutine:	PLAYER
Screen subroutine:	SCREEN.SRC
Music data:	filename.MUS

Use DOS (by inserting the General Utility Diskette into disk drive 1 and typing DOS) to copy the AUTORUN.SYS file to your Personal Diskette.

Now insert the General Utility Diskette into your drive and type the following:

```
NEW
ENTER     "D:PAINT"
ENTER     "D:ANIMATE:
ENTER     "D:MUSIC:
```

Insert your Personal Diskette into the disk drive and type the following:

```
ENTER     "D:PLAYER"
ENTER     "D:SCREEN.SRC"  (if you used SCREEN MAKER)

SAVE      "D:SHOW"
```

The next step is to teach your players their animation sequence. When teaching the players, do not use music -- only put up your background painting by typing the following:

```
10 FILE$="filename.PIC":GOSUB PAINT
20 PATH=1:GOSUB TEACH
```

And, if you have more than one animation sequence...

```
30 FILE$="filename.PIC":GOSUB PAINT  (if you have another painting)
40 PATH=2:GOSUB TEACH
```

When you have finished teaching your players, type the following:

```
LOAD "D:SHOW"

10 FILE$="filename.MUS":GOSUB MUSIC
           (This loads your music data.  Any time you want to load a
           different tune, use this same statement with a different
           filename.)

20 WINDOW=2:GOSUB MYSCREEN
30 PRINT #6; "..."
           (Use this only if you used SCREEN MAKER.)

40 FILE$="filename.PIC":GOSUB PAINT
           (This loads your painting.  Any time you want to load a
           different picture, use this same statement with a different
           filename.)

50 GOSUB PLAY
           (This starts your tune from the beginning.  GOSUB HALT will
           stop it and GOSUB RESUME will restart it from where it
           stopped.)

60 PATH=1:GOSUB ANIMATE

70 ...
           (You can continue with different pictures, different names,
           and different animation sequences by repeating the same
           processes as used above.)
```

When you have finished your program, be sure to save it by typing:

```
SAVE "D:SHOW"
```

Before your program can play music, you must turn your computer off and then back on so that the AUTORUN.SYS file gets loaded into memory. Then type: RUN "D:SHOW".

APPENDIX A

PROGRAMS ON THE BASIC UTILITY DISK

```
  NEAT        006
  CLOWN       006
  CUBE        008
  BOX         005
  COLOR       004
  CLOWN   DAT 024
  SOUNDS      007
  PRINTS      007
  DOS     SYS 039
  DUP     SYS 042
  ANARROW     005
  TUNE        011
  CHOMP       012
  MSHUT   GR  003
  MOPEN   GR  003
  BOAT    GR  003
  TREE    GR  003
  ROCKET  GR  003
  CHMAIN      003
  JOYSTK1     004
  JSTICK      005
  USINGAND    006
  SHIP        016
  SUN     GR  005
  INTRO   TXT 003
  LEAVING TXT 003
  BEACH   TXT 004
  STORM   TXT 002
  PHONHOMETXT 004
  ENDROPE TXT 003
  SUNSHINETXT 002
  CONTINUETXT 003
  VACATION    055
  FORTUNE     005
  PASSWORD    004
  WHOAMI      023
  SOUNDARY    012
  NUMPUZL     018
  JOYARROW    011
  ARTSHOW     011
314 FREE SECTORS
```

```
  NEAT        006
  CLOWN       006
  CUBE        008
  BOX         005
  COLOR       004
  CLOWN   DAT 024
  SOUNDS      007
  PRINTS      007
  DOS     SYS 039
  DUP     SYS 042
  ANARROW     005
  TUNE        011
  CHOMP       012
  MSHUT   GR  003
  MOPEN   GR  003
  BOAT    GR  003
  TREE    GR  003
  ROCKET  GR  003
  CHMAIN      003
  JOYSTK1     004
  JSTICK      005
  USINGAND    006
  SHIP        016
  SUN     GR  005
  INTRO   TXT 003
  LEAVING TXT 003
  BEACH   TXT 004
  STORM   TXT 002
  PHONHOMETXT 004
  ENDROPE TXT 003
  SUNSHINETXT 002
  CONTINUETXT 003
  VACATION    055
  FORTUNE     005
  PASSWORD    004
  WHOAMI      023
  SOUNDARY    012
  NUMPUZL     018
  JOYARROW    011
  ARTSHOW     011
314 FREE SECTORS
```

```
100 REM *    ART SHOW
105 REM *
110 MENU=900:REM MENU LINE NUMBER
115 GRAPHICS 7:COLOR 3
120 REM *
125 REM *****      MAIN LOOP    *****
130 GOSUB MENU
140 INPUT RESPONSE
150 IF RESPONSE<1 OR RESPONSE>4 THEN 140
160 IF RESPONSE=1 THEN RESTORE 500
170 IF RESPONSE=2 THEN RESTORE 600
180 IF RESPONSE=3 THEN RESTORE 700
190 IF RESPONSE=4 THEN RESTORE 800
200 REM *
210 REM *****      DRAW ROUTINE      *****
220 REM *
230 READ X,Y
240 PLOT X,Y:REM PICTURE START POINT
250 READ X,Y:REM GET DRAWTO DATA
260 IF X=-1 THEN 130:REM THE FLAG?
270 DRAWTO X,Y
280 GOTO 250:REM GET MORE DATA
290 REM *
500 REM *****      MOUNTAIN      *****
510 REM *
520 DATA 0,26,12,20,20,23,30,18,35,12,42,13,45,10,58,6
530 DATA 62,3,70,1,82,3,90,8,102,20,112,26,120,23
540 DATA 130,38,135,36,150,43,-1,-1
550 REM *
600 REM *****      BARN        *****610 REM *
610 REM *
620 DATA 43,50,43,46,47,46,47,50,40,50
630 DATA 40,44,45,41,50,44,50,50,40,50,-1,-1
640 REM *
700 REM *****      STAR      *****
710 REM *
720 DATA 128,10,127,11,126,11,127,12,126,13,127,13,128,14,129,13
730 DATA 130,13,129,12,130,11,129,11,128,11,128,14,-1,-1
740 REM *
800 REM *****      HORSE      *****
810 REM *
820 DATA 52,47,54,46,54,45,54,50,54,48,57,48,58,47,57,48,57,50,-1,-1
900 REM *
910 REM *****  MENU      *****
920 REM *
930 PRINT
940 PRINT "1.  MOUNTAIN      3.  STAR"
950 PRINT "2.  BARN          4.  HORSE"
960 PRINT "WHICH PICTURE (1, 2, 3, OR 4) ";
970 RETURN
```

```
10 REM CONTROLLED ARROW
20 GRAPHICS 7
30 PRINT "JOYSTCK IN PORT 1: CONTROLS DIRECTION.";
40 PRINT "             TRIGGER BUTTON CHANGES HUE.";
50 PRINT "PADDLE IN PORT 2:  CONTROLS LUMINANCE.";
60 X=15:Y=40:HUE=0:LUM=14
70 SETCOLOR 0,HUE,LUM
80 COLOR 1:GOSUB 300
90 COLOR 0:GOSUB 300
100 IF STRIG(0)=0 THEN HUE=HUE+1
110 IF HUE>15 THEN HUE=0
120 LUM=INT(PADDLE(2)/16)
130 IF STICK(0)=14 THEN Y=Y-1
140 IF STICK(0)=6 THEN X=X+1:Y=Y-1
150 IF STICK(0)=7 THEN X=X+1
160 IF STICK(0)=5 THEN X=X+1:Y=Y+1
170 IF STICK(0)=13 THEN Y=Y+1
180 IF STICK(0)=9 THEN X=X-1:Y=Y+1
190 IF STICK(0)=11 THEN X=X-1
200 IF STICK(0)=10 THEN X=X-1:Y=Y-1
210 GOTO 70
220 STOP
300 PLOT X,Y
310 DRAWTO X-15,Y
320 DRAWTO X-15,Y+2
330 DRAWTO X,Y+2
340 PLOT X+2,Y+1
350 DRAWTO X-6,Y-3
360 PLOT X+2,Y+1
370 DRAWTO X-6,Y+5
380 RETURN
390 END
400 REM *
410 REM *
420 REM *
430 REM *
440 REM *
450 REM *        JOYSTICK                      PADDLE
460 REM *
470 REM *
480 REM *           14
490 REM *
500 REM *       10        6
510 REM *
520 REM *   11      15       7          228            1
530 REM *
540 REM *        9        5
550 REM *
560 REM *           13
```

```
100 REM *     SOUND ARRAY
110 REM *
120 REM *
130 REM *  INITIALIZE VARIABLES AND ARRAY
140 REM *
150 DIM TUNE(100)
160 XNOTE=0
165 REM * ASSIGN LABELS TO LINE NUMBERS
170 MENU=300
180 VALUES=500
190 PLAY=700
200 NUMBERS=900
210 REM *
220 REM *     MAIN LOOP
230 REM *
240 GOSUB MENU
250 INPUT RESPONSE
260 IF RESPONSE=1 THEN GOSUB VALUES
270 IF RESPONSE=2 THEN GOSUB PLAY
280 IF RESPONSE=3 THEN GOSUB NUMBERS
290 GOTO 240:REM REPEAT MAIN LOOP
300 REM *
310 REM *     MENU
320 REM *
330 PRINT
340 PRINT "WOULD YOU LIKE TO:"
350 PRINT "     1.  TYPE IN A TUNE."
360 PRINT "     2.  PLAY YOUR TUNE."
370 PRINT "     3.  LIST THE NOTES."
380 PRINT
390 PRINT "TYPE IN A NUMBER";
400 PRINT :REM INPUT IN MAIN LOOP
410 RETURN
500 REM *
510 REM *     INPUT VALUES FOR NOTES
520 REM *
530 PRINT "  TYPE IN NUMBERS BETWEEN 0"
540 PRINT "  AND 255 TO BE THE NOTES"
550 PRINT "  OF A TUNE.  TYPE ONE NOTE"
560 PRINT "  PER ?.  WHEN YOU ARE FINISHED,"
570 PRINT "  TYPE A -1 FOR THE LAST NOTE."
580 INPUT PITCH
590 IF PITCH>255 OR PITCH<-1 THEN 580
600 REM *  MINUS ONE IS A FLAG FOR THE END OF THE DATA
610 IF PITCH=-1 THEN NUMNOTES=XNOTE:RETURN
620 XNOTE=XNOTE+1:REM NOTES COUNTER
630 TUNE(XNOTE)=PITCH
640 GOTO 580
700 REM *
710 REM *     PLAY TUNE
720 REM *
730 FOR XNOTE=1 TO NUMNOTES
740 SOUND 0,TUNE(XNOTE),10,10
750 FOR DELAY=1 TO 10:NEXT DELAY
760 NEXT XNOTE
770 SOUND 0,0,0,0
780 RETURN
900 REM *
910 REM *     LIST NOTES
920 REM *
930 FOR XNOTE=1 TO NUMNOTES
940 PRINT "TUNE(";XNOTE;")";"  ";TUNE(XNOTE)
950 NEXT XNOTE
960 RETURN
```

```
100 REM *        TUNE ARRAY
110 REM *
120 DIM PITCH(50),DISTORT(50),LOUD(50),TIME(50)
130 INIT=500:REM INITIALIZATION LINE#
140 PLAY=300:REM PLAY TUNE ROUTINE
150 MAXNOTES=11
200 REM *
210 REM *****  MAIN LOOP  *****
220 REM *
230 GOSUB INIT
240 START=1:FINISH=5:GOSUB PLAY
250 START=6:FINISH=11:GOSUB PLAY
260 START=1:FINISH=4:GOSUB PLAY
270 END
300 REM
310 REM *****  PLAY  *****
320 REM *
330 REM *  PLAYS A SEQUENCE OF NOTES USING DATA ARRAYS.
340 REM *  INDICES DETERMINED BY VALUES OF START AND
350 REM *  FINISH IN MAIN LOOP
360 REM *
370 FOR XNOTE=START TO FINISH
380 SOUND 0,PITCH(XNOTE),DISTORT(XNOTE),LOUD(XNOTE)
390 FOR DELAY=1 TO TIME(XNOTE):NEXT DELAY
400 NEXT XNOTE
410 RETURN
420 REM *
500 REM *****  INIT ARRAY  *****
510 REM *
520 FOR FILL=1 TO MAXNOTES
530 READ PITCH,DISTORT,LOUD,TIME
540 PITCH(FILL)=PITCH:DISTORT(FILL)=DISTORT:LOUD(FILL)=LOUD:TIME(FILL)=TIME
550 NEXT FILL
560 RETURN
570 DATA 121,10,10,40,91,10,10,37,0,0,0,3,91,10,10,40,108,10,10,28
580 DATA 0,0,0,2,108,10,10,10,91,10,10,30,108,10,10,10,121,10,10,80,0,0,0,0
```

```
100 REM *     BATTLE SHIP
110 REM *
120 REM *
130 REM *  INITIALIZE VARIABLES
140 REM *
150 MAXLOCATIONS=4
160 DIM BOARD(MAXLOCATIONS,MAXLOCATIONS)
170 COLUMN=0:ROW=0
175 REM ASSIGN NAMES TO SUBROUTINE LINE NUMBERS
180 INITMATRIX=500
190 PLACESHIP=700
200 PLAY=900
210 WIN=1300
300 REM *
310 REM *     MAIN LOOP
320 REM *
330 GOSUB `INITMATRIX
340 GOSUB PLACESHIP
350 GOSUB PLAY
360 END
500 REM *
510 REM *     INIT MATRIX
520 REM *
530 FOR ROW=1 TO MAXLOCATIONS
540 FOR COLUMN=1 TO MAXLOCATIONS
550 BOARD(ROW,COLUMN)=0
560 PRINT BOARD(ROW,COLUMN)
570 NEXT COLUMN
580 NEXT ROW
590 RETURN
700 REM *
710 REM *     PLACE SHIP
720 REM *
730 SHIPROW=INT(RND(0)*4)+1:REM RANDOM NUMBER
740 SHIPCOL=INT(RND(0)*4)+1:REM BETWEEN 1 AND 4
750 BOARD(SHIPROW,SHIPCOL)=1:REM PLACE SHIP IN RANDOM LOCATION
760 RETURN
900 REM *
910 REM *     PLAY
920 REM *
930 GRAPHICS 2
940 POSITION 0,0
950 PRINT #6;"       columns"
960 FOR NUMBER=1 TO 4
970 POSITION 2,NUMBER*2:REM ROW COORDINATES
980 PRINT #6;NUMBER
990 POSITION NUMBER*4,1:REM COLUMN COORDINATES
1000 PRINT #6;NUMBER
1010 NEXT NUMBER
```

```
1020 POSITION 0,3:PRINT #6;"R"
1030 POSITION 0,4:PRINT #6;"O"
1040 POSITION 0,5:PRINT #6;"W"
1050 POSITION 0,6:PRINT #6;"S"
1060 PRINT "  TYPE IN THE COORDINATES OF"
1070 PRINT "  YOUR GUESS.  THE NUMBER MUST"
1080 PRINT "  BE BETWEEN 1 AND 4."
1090 PRINT "ROW: ";
1100 INPUT ROWGUESS
1110 IF ROWGUESS<1 OR ROWGUESS>4 THEN 1090
1120 PRINT "COLUMN: ";
1130 INPUT COLGUESS
1140 IF COLGUESS<1 OR COLGUESS>4 THEN 1120
1150 IF BOARD(ROWGUESS,COLGUESS)=1 THEN GOSUB WIN:RETURN
1160 POSITION COLGUESS*4,ROWGUESS*2:REM PUT * ON BOARD
1170 PRINT #6;"*"
1180 PRINT :PRINT "TRY AGAIN"
1190 GOTO 1090
1300 REM *
1310 REM *      WIN
1320 REM *
1330 PRINT :PRINT "YOU FOUND IT!"
1340 FOR COUNT=1 TO 10
1350 POSITION COLGUESS*4,ROWGUESS*2
1360 PRINT #6;" ";:REM ERASE *
1370 FOR DELAY=1 TO 75:NEXT DELAY
1380 POSITION COLGUESS*4,ROWGUESS*2
1390 PRINT #6;"*";:REM FLASH *
1400 FOR DELAY=1 TO 75:NEXT DELAY
1410 NEXT COUNT
1420 RETURN
```

```
10 REM ***  Number Pattern Puzzle ***
20 DIM ANS$(1),MORE$(1)
30 ? "}":? :?
40 ? "This is a number puzzle.  Try"
50 ? "to figure out what the next"
60 ? "number will be in each sequence"
70 ? "These are the sequences:":?
80 ? "A.  1, 3, 5, 7, 11, 13, 17, ___"
90 ? "B.  40, 51, 62, 73, 84, 95, ___"
100 ? "C.  1, 1, 2, 3, 5, 8, 13, 21, ___"
110 ? "D.  3, 3, 5, 4, 4, 3, 5, 5, ___":?
120 ? "What sequence would you like"
130 ? "to try? (Type a letter.)";
140 INPUT ANS$
150 IF ANS$="A" THEN GOSUB 5000
160 IF ANS$="B" THEN GOSUB 5100
170 IF ANS$="C" THEN GOSUB 5200
180 IF ANS$="D" THEN GOSUB 5300
200 ? :? "Do you want to try another"
210 ? "sequence? (Type Y or N)";
220 INPUT MORE$
230 IF MORE$="Y" THEN ? "}":GOTO 70
240 END
5000 REM ****prime number sequence***
5005 REM ****                     ***
5010 ? "}":? :? :REM * clears screen  *
5020 ? "Type a number to finish this"
5025 ? "sequence. You get three chances.":?
5030 ? "1, 3, 5, 7, 11, 13, 17, ____"
5040 FOR COUNT=1 TO 3
5050 ? :? "NUMBER";
5060 INPUT NUM
5070 IF NUM=19 THEN ? "That's correct.":RETURN
5080 ? "It's not ";NUM;".  These are prime numbers."
5090 NEXT COUNT
5095 RETURN
```

```
5100 REM ****add eleven sequence*****
5105 REM ****                   *****
5110 ? "}":? :? :REM * clears screen  *
5120 ? "Type a number to finish this"
5125 ? "sequence. You get three chances.":?
5130 ? "40, 51, 62, 73, 84, 95, ____"
5140 FOR COUNT=1 TO 3
5150 ? :? "NUMBER";
5160 INPUT NUM
5170 IF NUM=106 THEN ? "That's correct.":RETURN
5180 ? "It's not ";NUM;"."
5190 NEXT COUNT
5195 RETURN
5200 REM ****Fibonacci sequence******
5205 REM ****                 ******
5210 ? "}":? :? :REM * clears screen  *
5220 ? "Type a number to finish this"
5225 ? "sequence. You get three chances.":?
5230 ? "1, 1, 2, 3, 5, 8, 13, 21, ____"
5240 FOR COUNT=1 TO 3
5250 ? :? "NUMBER";
5260 INPUT NUM
5270 IF NUM=34 THEN ? "That's correct.":RETURN
5280 ? "It'S not ";NUM;".  It's a Fibonacci."
5290 NEXT COUNT
5295 RETURN
5300 REM **number of letters in nums*
5305 REM ***                      ***
5310 ? "}":? :? :REM * clears screen  *
5320 ? "Type a number to finish this"
5325 ? "sequence. You get three chances.":?
5330 ? "3, 3, 5, 4, 4, 3, 5, 5, ____"
5340 FOR COUNT=1 TO 3
5350 ? :? "NUMBER";
5360 INPUT NUM
5370 IF NUM=4 THEN ? "That's correct.":RETURN
5380 ? "It's not ";NUM;"."
5390 NEXT COUNT
5395 RETURN
```

```
10 REM *Who Am I-Names in Computing*
20 DIM CORRECTANS$(1),USERANS$(1),R$(1)
30 GOSUB 2600:REM **   Directions **
40 GOSUB 2100:REM **    Babbage   **
50 GOSUB 2800:REM **    Answers   **
60 CORRECTANS$="E"
70 GOSUB 2700:REM ** Answer Input **
80 GOSUB 29000:REM **  Wait Loop  **
90 GOSUB 2500:REM **   Hollerith **
100 GOSUB 2800:REM **  Answers    **
110 CORRECTANS$="B"
120 GOSUB 2700:REM ** Answer Input**
130 GOSUB 29000:REM **  Wait Loop **
140 GOSUB 2200:REM **     Ada     **
150 GOSUB 2800:REM **   Answers   **
160 CORRECTANS$="C"
170 GOSUB 2700:REM ** Answer Input**
180 GOSUB 29000:REM **  Wait Loop **
190 GOSUB 2300:REM **   Pascal    **
200 GOSUB 2800:REM **   Answers   **
210 CORRECTANS$="A"
220 GOSUB 2700:REM ** Answer Input**
230 GOSUB 29000:REM ** Wait Loop  **
240 GOSUB 2400:REM **  Boole      **
250 GOSUB 2800:REM **   Answers   **
260 CORRECTANS$="D"
270 GOSUB 2700:REM ** Answer Input**
280 GOSUB 29000:REM ** Wait Loop  **
290 END
2100 REM ***    Charles Babbage    ***
2110 ? "}":PRINT
2120 ? "An English mathematician and"
2130 ? "inventor who is often called"
2140 ? "the Father of Computing.  He"
2150 ? "said, 'I am thinking that all"
2160 ? "those tables might be"
2170 ? "calculated by machinery.'"
2180 ? :?
2190 RETURN
2200 REM ******       ADA       *****
2210 ? "}":PRINT
2220 ? "An exceptional English mathematician"
2230 ? "who is credited with being the first"
2240 ? "person to make the statement that"
2250 ? "computers can do only what you"
2260 ? "program them to do.  Wrote about"
2270 ? "Babbage's Analytical Engine."
2280 PRINT :PRINT
2290 RETURN
```

```
2300 REM ***     Blaise Pascal       ***
2310 ? "}":PRINT
2320 ? "A French mathematician who was"
2330 ? "the first person to invent a"
2340 ? "significant calculating "
2350 ? "machine."
2360 PRINT :PRINT
2370 RETURN
2400 REM ***           Boole           ***
2410 ? "}":PRINT
2420 ? "An English logician.  The"
2430 ? "pioneer of modern symbolic logic."
2440 PRINT :PRINT
2450 RETURN
2500 REM ***     Hollerith        ***
2510 ? "}":?
2520 ? "An American inventor.  The"
2560 ? "first to do a practical"
2570 ? "implementation of punched cards."
2580 PRINT :PRINT
2590 RETURN
2600 REM **  Who Am I - Directions **
2610 ? "}":? :?
2620 ? "         WHO AM I?"
2625 ?
2630 ? "After you read the description"
2640 ? "of the person, choose your"
2650 ? "answer from the names given."
2660 ? "Type in the letter of the"
2670 ? "correct answer.  Each of the"
2680 ? "names is famous in computing."
2685 ? "Press RETURN when you are "
2690 ? "ready to begin."
2695 INPUT R$
2699 RETURN
2700 REM ***  Asks for answer    ***
2710 REM *input with correct answer*
2720 ? :? "Who am I?  Type a letter";
2730 INPUT USERANS$
2740 IF USERANS$=CORRECTANS$ THEN ? "That's correct.":RETURN
2750 ? "That's not my name.  The"
2760 ? "correct answer is ";CORRECTANS$;"."
2770 RETURN
2800 REM ***      Answers        ***
2810 ? :? :?
2820 ? "        A.  Pascal"
2830 ? "        B.  Hollerith"
2840 ? "        C.  Ada"
2850 ? "        D.  Boole"
2860 ? "        E.  Babbage"
2870 RETURN
29000 REM ** Delay Loop  **
29010 FOR WAIT=1 TO 500:NEXT WAIT
29020 RETURN
```

```
10 REM * Self destructive program *
20 ? "}":REM * Clears Screen *
30 DIM WORD$(25),PASSWORD$(25)
40 ? "If you want to read my message,"
50 ? "you must give me the password."
60 ? "If you give me the wrong word,"
70 ? "the program destroys itself."
80 INPUT WORD$
90 PASSWORD$="ATARI"
100 IF WORD$<>PASSWORD$ THEN NEW
110 ? :? "Congratulations!  You guessed"
120 ? "the pàssword."
130 END
```

```
10 REM Fortune teller
20 PRINT "}":REM Clear screen
30 PRINT :PRINT "I will tell you your fortune."
40 PRINT "Let's see...":PRINT
50 NUM=INT(3*RND(0))
60 FOR WAIT=1 TO 1000:NEXT WAIT
70 PRINT "+ + + + + + + + + + + + +"
80 PRINT "You will become very ";
90 IF NUM=0 THEN PRINT "rich."
100 IF NUM=1 THEN PRINT "poor."
110 IF NUM=2 THEN PRINT "powerful."
120 NUM=INT(3*RND(0))
130 FOR WAIT=1 TO 750:NEXT WAIT
140 PRINT "You will also be very ";
150 IF NUM=0 THEN PRINT "happy."
160 IF NUM=1 THEN PRINT "famous."
170 IF NUM=2 THEN PRINT "popular."
180 PRINT "+ + + + + + + + + + + + + +"
190 GOTO 30
```

```
10 REM *********MAIN PROGRAM*********
50 GOSUB 10300:REM *   Title Page    *
60 GOSUB 10100:REM *  Author Page    *
70 GOSUB 1100:REM *  Introduction   *
80 GOSUB 29000:REM *   Wait Loop    *
90 GOSUB 10000:REM *Going in circles*
100 GOSUB 1200:REM *Leaving the city*
110 GOSUB 20500:REM *  Train sound  *
120 GOSUB 1300:REM * Arrive at beach*
130 GOSUB 20900:REM * Ocean sound   *
140 GOSUB 1400:REM *  The Storm     *
150 GOSUB 29000:REM *   Wait Loop   *
160 GOSUB 10600:REM *  Rain graphic *
170 GOSUB 1700:REM *    Rain text   *
180 GOSUB 29000:REM *   Wait Loop   *
190 GOSUB 10600:REM *  Rain graphic *
200 GOSUB 1700:REM *    Rain text   *
210 GOSUB 29000:REM *   Wait Loop   *
220 GOSUB 10600:REM *  Rain graphic *
230 GOSUB 1500:REM *   Phone Home   *
240 GOSUB 20300:REM *  Busy Signal  *
250 GOSUB 1800:REM *  End of Rope   *
260 GOSUB 20100:REM * Chirping birds*
270 GOSUB 1900:REM *  Sunshine text *
280 GOSUB 29000:REM *   Wait Loop   *
290 GOSUB 10200:REM *  Sun graphic  *
300 GOSUB 2000:REM *   Noend text   *
310 GOSUB 29000:REM *   Wait Loop   *
350 END :REM * End of Main Program  *
360 REM
370 REM
380 REM
1100 REM ******Introduction**********
1105 GRAPHICS 1+16
1110 ? #6
1115 ? #6
1120 ? #6
1125 ? #6;"A MAN NAMED FRED"
1127 ? #6
1130 ? #6;"WAS VERY BORED"
1135 ? #6
1140 ? #6;"WITH LIFE.  IT"
1145 ? #6
1150 ? #6;"SEEMED LIKE ALL HE"
1155 ? #6
1160 ? #6;"EVER DID WAS GO"
1165 ? #6
1170 ? #6;"AROUND IN CIRCLES."
1180 RETURN
```

```
1200 REM *****Leaving the city*******
1205 GRAPHICS 1+16
1210 ? #6
1215 ? #6
1220 ? #6
1225 ? #6;"ONE DAY HE "
1227 ? #6
1230 ? #6;"DECIDED TO LEAVE"
1235 ? #6
1240 ? #6;"THE BIG CITY."
1245 ? #6
1250 ? #6;"HE GOT ON A"
1255 ? #6
1265 ? #6;"TRAIN......."
1270 RETURN
1300 REM *****Arrival at beach*******
1302 GRAPHICS 1+16
1305 ? #6
1310 ? #6;"......AND WENT TO"
1315 ? #6
1320 ? #6;"THE BEACH FOR A"
1325 ? #6
1330 ? #6;"VACATION. THE DAY"
1335 ? #6
1340 ? #6;"HE ARRIVED, IT WAS"
1345 ? #6
1350 ? #6;"SUNNY AND WARM."
1355 ? #6
1360 ? #6;"THE SOUND OF THE"
1365 ? #6
1370 ? #6;"OCEAN WAS VERY"
1375 ? #6
1380 ? #6;"CALMING TO HIS"
1385 ? #6
1390 ? #6;"NERVES."
1395 RETURN
1400 REM **********Storm*************
1410 GRAPHICS 2+16
1425 ? #6
1440 ? #6
1445 ? #6;"H O W E V E R....."
1450 ? #6
1455 ? #6;"THAT NIGHT A STORM"
1460 ? #6
1465 ? #6;"CAME UP AND IT"
1470 ? #6
1475 ? #6;"RAINED....."
1480 RETURN
```

```
1500 REM ********Phone home**********
1501 GRAPHICS 1+16
1505 ? #6
1510 ? #6;"HE DECIDED TO "
1515 ? #6
1520 ? #6;"PHONE HOME TO SEE"
1525 ? #6
1530 ? #6;"IF THE WEATHER"
1535 ? #6
1540 ? #6;"WAS ANY BETTER "
1545 ? #6
1550 ? #6;"THERE.  BUT SINCE"
1555 ? #6
1560 ? #6;"HIS CHILDREN WERE"
1565 ? #6
1570 ? #6;"ALWAYS ON THE "
1575 ? #6
1580 ? #6;"PHONE, ALL HE GOT"
1585 ? #6
1590 ? #6;"WAS A BUSY SIGNAL."
1595 RETURN
1700 REM ******And it rained*********
1710 GRAPHICS 2+16
1720 ? #6
1730 ? #6
1740 ? #6
1750 ? #6
1755 ? #6
1760 ? #6;"  AND IT RAINED..."
1770 RETURN
1800 REM *****End of his rope********
1805 GRAPHICS 1+16
1810 ? #6
1815 ? #6
1820 ? #6
1825 ? #6
1830 ? #6
1840 ? #6;"JUST AS HE WAS AT"
1845 ? #6
1850 ? #6;"THE END OF HIS "
1855 ? #6
1860 ? #6;"ROPE AND READY TO"
1865 ? #6
1870 ? #6;"RETURN HOME, THE"
1875 ? #6
1880 ? #6;"BIRDS BEGAN TO"
1885 ? #6
1890 ? #6;"SING..."
1895 RETURN
```

```
1900 REM *********Sunshine***********
1902 GRAPHICS 2+16
1910 ? #6
1920 ? #6
1930 ? #6
1950 ? #6
1960 ? #6
1980 ? #6;"THE SUN CAME OUT...."
1990 RETURN
2000 REM *****To be continued******
2010 GRAPHICS 2+16
2020 ? #6
2030 ? #6;".  ....AND HE...."
2040 ? #6
2050 ? #6;" (TO BE CONTINUED)"
2060 ? #6
2065 ? #6
2070 ? #6;"...................."
2080 ? #6;"YOU FINISH THE STORY"
2085 ? #6;"...................."
2095 RETURN
10000 REM *****Going in circles*****
10010 GRAPHICS 7+16:COLOR 2
10020 FOR COUNTER=1 TO 50
10030 Z=Z+0.5
10040 X=SIN(Z)*25:Y=COS(Z)*22
10050 PLOT X+80,Y+45
10060 NEXT COUNTER
10070 RETURN
10100 REM *******Author Page*********
10110 GRAPHICS 2+16
10120 POSITION 4,2:PRINT #6;"*************"
10130 POSITION 4,3:PRINT #6;"*           *"
10135 POSITION 4,4:PRINT #6;"*    BY     *"
10140 POSITION 4,5:PRINT #6;"*           *"
10150 POSITION 4,6:PRINT #6;"* YOUR NAME *"
10155 POSITION 4,7:PRINT #6;"*           *"
10160 POSITION 4,8:PRINT #6;"*************"
10170 GOSUB 29000:REM *  Wait Loop   *
```

```
10200 REM ***********Sun***********
10205 GRAPHICS 3+16
10210 ? #6;"                =     "
10215 ? #6;"      =        =      "
10220 ? #6;"       =      =       "
10225 ? #6;"        =    =        "
10230 ? #6;"         ====         "
10235 ? #6;"        ======        "
10240 ? #6;"       ======== = = = "
10245 ? #6;" = = = ========       "
10250 ? #6;"       ========       "
10255 ? #6;"        ======        "
10260 ? #6;"         ==== =       "
10270 ? #6;"         =     =      "
10275 ? #6;"        =       =     "
10280 ? #6;"       =         =    "
10285 ? #6;"      =               "
10290 GOSUB 29000:REM *  Wait Loop   *
10295 RETURN
10300 REM ********Title page********
10310 GRAPHICS 2+16:COLOR 2
10315 PRINT #6:PRINT #6:PRINT #6:PRINT #6
10320 PRINT #6;"         A"
10325 PRINT #6;"      VACATION"
10330 PRINT #6;"       STORY"
10340 PLOT 1,1
10345 DRAWTO 19,1
10350 DRAWTO 19,9
10355 DRAWTO 1,9
10360 DRAWTO 1,1
10365 GOSUB 29000:REM *  Wait Loop   *
10370 RETURN
10600 REM *********Rain**************
10605 FOR LOOP=1 TO 3
10610 GRAPHICS 3+16
10620 FOR COUNTER=1 TO 84
10630 PRINT #6,"+";
10640 NEXT COUNTER
10650 GRAPHICS 0
10660 NEXT LOOP
10670 RETURN
```

```
20100 REM *******Chirping Birds******
20110 FOR LOOP=1 TO 4
20120 FOR COUNT=1 TO 5
20130 FOR PITCH=1 TO 15
20140 SOUND 2,PITCH,10,8
20150 NEXT PITCH
20160 NEXT COUNT
20170 NEXT LOOP
20180 SOUND 2,0,0,0
20190 RETURN
20300 REM ***Telephone Busy Signal***
20305 FOR RINGS=1 TO 9
20310 SOUND 2,40,6,10
20320 FOR WAIT=1 TO 50:NEXT WAIT
20330 SOUND 2,0,0,0
20340 FOR WAIT=1 TO 25:NEXT WAIT
20350 NEXT RINGS
20360 RETURN
20500 REM *****Steam Locomotive******
20510 FOR LOOP=1 TO 25
20520 FOR LOUD=10 TO 0 STEP -1
20530 SOUND 0,15,0,LOUD
20540 NEXT LOUD
20550 NEXT LOOP
20560 SOUND 0,0,0,0
20570 RETURN
20900 REM ***********Ocean***********
20910 FOR LOOP=1 TO 2
20920 FOR PITCH=0 TO 12
20930 SOUND 0,PITCH,8,6
20940 FOR WAIT=1 TO 5:NEXT WAIT
20950 NEXT PITCH
20960 FOR PITCH=12 TO 0 STEP -1
20970 SOUND 0,PITCH,8,4
20975 FOR WAIT=1 TO 17:NEXT WAIT
20980 NEXT PITCH
20985 NEXT LOOP
20990 SOUND 0,0,0,0
20995 RETURN
29000 REM ********Wait loop**********
29010 FOR WAIT=1 TO 500:NEXT WAIT
29020 RETURN
```

```
2000 REM *****To be continued******
2010 GRAPHICS 2+16
2020 ? #6
2030 ? #6;"  ....AND HE...."
2040 ? #6
2050 ? #6;" (TO BE CONTINUED)"
2060 ? #6
2065 ? #6
2070 ? #6;"...................."
2080 ? #6;"YOU FINISH THE STORY"
2085 ? #6;"...................."
2095 RETURN
```

```
1900 REM *********Sunshine***********
1902 GRAPHICS 2+16
1910 ? #6
1920 ? #6
1930 ? #6
1950 ? #6
1960 ? #6
1980 ? #6;"THE SUN CAME OUT...."
1990 RETURN
```

```
1800 REM *****End of his rope*********
1805 GRAPHICS 1+16
1810 ? #6
1815 ? #6
1820 ? #6
1825 ? #6
1830 ? #6
1840 ? #6;"JUST AS HE WAS AT"
1845 ? #6
1850 ? #6;"THE END OF HIS "
1855 ? #6
1860 ? #6;"ROPE AND READY TO"
1865 ? #6
1870 ? #6;"RETURN HOME, THE"
1875 ? #6
1880 ? #6;"BIRDS BEGAN TO"
1885 ? #6
1890 ? #6;"SING..."
1895 RETURN
```

```
1500 REM ********Phone home***********
1501 GRAPHICS 1+16
1505 ? #6
1510 ? #6;"HE DECIDED TO "
1515 ? #6
1520 ? #6;"PHONE HOME TO SEE"
1525 ? #6
1530 ? #6;"IF THE WEATHER"
1535 ? #6
1540 ? #6;"WAS ANY BETTER "
1545 ? #6
1550 ? #6;"THERE.  BUT SINCE"
1555 ? #6
1560 ? #6;"HIS CHILDREN WERE"
1565 ? #6
1570 ? #6;"ALWAYS ON THE "
1575 ? #6
1580 ? #6;"PHONE, ALL HE GOT"
1585 ? #6
1590 ? #6;"WAS A BUSY SIGNAL."
1595 RETURN
```

```
1400 REM **********Storm*************
1410 GRAPHICS 2+16
1425 ? #6
1440 ? #6
1445 ? #6;"H O W E V E R....."
1450 ? #6
1455 ? #6;"THAT NIGHT A STORM"
1460 ? #6
1465 ? #6;"CAME UP AND IT"
1470 ? #6
1475 ? #6;"RAINED....."
1480 RETURN
```

```
1300 REM *****Arrival at beach*******
1302 GRAPHICS 1+16
1305 ? #6
1310 ? #6;"......AND WENT TO"
1315 ? #6
1320 ? #6;"THE BEACH FOR A"
1325 ? #6
1330 ? #6;"VACATION. THE DAY"
1335 ? #6
1340 ? #6;"HE ARRIVED, IT WAS"
1345 ? #6
1350 ? #6;"SUNNY AND WARM."
1355 ? #6
1360 ? #6;"THE SOUND OF THE"
1365 ? #6
1370 ? #6;"OCEAN WAS VERY"
1375 ? #6
1380 ? #6;"CALMING TO HIS"
1385 ? #6
1390 ? #6;"NERVES."
1395 RETURN
```

```
1200 REM *****Leaving the city*******
1205 GRAPHICS 1+16
1210 ? #6
1215 ? #6
1220 ? #6
1225 ? #6;"ONE DAY HE "
1227 ? #6
1230 ? #6;"DECIDED TO LEAVE"
1235 ? #6
1240 ? #6;"THE BIG CITY."
1245 ? #6
1250 ? #6;"HE GOT ON A"
1255 ? #6
1265 ? #6;"TRAIN......."
1270 RETURN
```

```
1100 REM ******Introduction**********
1105 GRAPHICS 1+16
1110 ? #6
1115 ? #6
1120 ? #6
1125 ? #6;"A MAN NAMED FRED"
1127 ? #6
1130 ? #6;"WAS VERY BORED"
1135 ? #6
1140 ? #6;"WITH LIFE.  IT"
1145 ? #6
1150 ? #6;"SEEMED LIKE ALL HE"
1155 ? #6
1160 ? #6;"EVER DID WAS GO"
1165 ? #6
1170 ? #6;"AROUND IN CIRCLES."
1180 RETURN
```

```
10200 REM ***********Sun***********
10205 GRAPHICS 3+16
10210 ? #6;"                =     "
10215 ? #6;"      =        =      "
10220 ? #6;"       =      =       "
10225 ? #6;"        =    =        "
10230 ? #6;"         ====         "
10235 ? #6;"        ======        "
10240 ? #6;"       ======== = = = "
10245 ? #6;" = = = ========       "
10250 ? #6;"       ========       "
10255 ? #6;"        ======        "
10260 ? #6;"         ==== =       "
10270 ? #6;"         =     =      "
10275 ? #6;"        =       =     "
10280 ? #6;"       =         =    "
10285 ? #6;"      =               "
10290 RETURN
```

```
10 REM ***   Using AND      ***
20 DIM ANS$(10),CORRECT$(1)
30 CORRECT$="N":COUNT=0
40 ? "}":?
50 ? "Who published 'On Computable"
60 ? "Numbers', one of the most important"
70 ? "papers in the foundations of"
80 ? "computer science?  You get three"
90 ? "chances to give the right answer.":?
100 INPUT ANS$
110 IF ANS$="Turing" THEN CORRECT$="Y"
120 COUNT=COUNT+1
130 ? "COUNT = ";COUNT
140 ? "CORRECT = ";CORRECT$:?
150 IF COUNT<>3 AND CORRECT$="N" THEN ? "Try again: ";:GOTO 100
160 IF CORRECT$="Y" THEN ? "Yes, it was Alan Matheson Turing.":GOTO 200
170 ? "This was a difficult question.  The"
180 ? "answer can be found on page 79 of"
190 ? "the book, 'The Making of the Micro'."
200 END
```

```
10 REM * Experimenting with a Joystick *
15 GRAPHICS 3+16
20 X=39:Y=19:C=1
30 XOLD=X:YOLD=Y
40 STK=STICK(0):IF STK<>15 THEN GOSUB 200
50 COLOR 0:PLOT XOLD,YOLD
60 COLOR C:PLOT X,Y
70 IF STRIG(0)=0 THEN GOSUB 300
80 GOTO 30
200 IF (STK=5) OR (STK=6) OR (STK=7) THEN X=X+1
210 IF (STK=9) OR (STK=10) OR STK=(11) THEN X=X-1
220 IF (STK=10) OR (STK=6) OR (STK=14) THEN Y=Y-1
230 IF (STK=9) OR (STK=13) OR (STK=5) THEN Y=Y+1
240 RETURN
300 C=C+1
310 IF C=4 THEN C=1
320 RETURN
```

```
10 REM * Practice with the joystick *
20 NUM=STICK(0)
30 IF NUM=15 THEN ? "RESTING"
40 IF NUM=14 THEN ? "GOING FORWARD"
50 IF NUM=13 THEN ? "GOING BACKWARD"
60 IF NUM=11 THEN ? "LEFT"
70 IF NUM=7 THEN ? "RIGHT"
80 IF NUM=10 THEN ? "UP & LEFT"
90 IF NUM=6 THEN ? "UP & RIGHT"
100 IF NUM=9 THEN ? "DOWN & LEFT"
110 IF NUM=5 THEN ? "DOWN & RIGHT"
120 ? "Joystick = ";NUM:?
130 FOR WAIT=1 TO 250:NEXT WAIT
140 GOTO 20
```

```
10 DIM LINE$(25),L$(25),B$(25)
20 PRINT "}":REM CLEAR SCREEN
30 POKE 752,1:REM TURN OFF CURSOR
100 POSITION 5,5
110 GOSUB 11300:REM TREE
120 POSITION 24,4
130 GOSUB 10900:REM ROCKET
140 POSITION 22,15
150 GOSUB 11200:REM BOAT
160 POSITION 6,16
170 GOSUB 11000:REM SHUT MOUTH
180 POSITION 6,16
190 GOSUB 11100:REM OPEN MOUTH
200 GOTO 160
10900 REM ******ROCKET******
10910 LINE$=""
10920 PRINT "       ";LINE$;
10930 PRINT "   |   ";LINE$;
10940 PRINT "
            ";LINE$;
10950 PRINT "      ";LINE$;
10960 PRINT "      ";LINE$;
10970 PRINT "      ";LINE$;
10980 PRINT "
            ";LINE$;
10985 PRINT " /`\ ";LINE$;
10990 PRINT "       ";
10995 RETURN
11000 REM *******SHUT MOUTH**********
11010 LINE$=""
11020 PRINT "       ";LINE$;
11030 PRINT "
            ";LINE$;
11040 PRINT "     ";LINE$;
11050 PRINT "    ";LINE$;
11060 PRINT "
            \;L/INE$;
11070 PRINT "
            ";LINE$;
11080 PRINT "       ";
11090 RETURN
```

```
11100 REM ********OPEN MOUTH*********
11110 LINE$=""
11120 PRINT "          ";LINE$;
11130 PRINT "
          ";LINE$;
11140 PRINT "       ";LINE$;
11150 PRINT "     ";LINE$;
11160 PRINT "

            ";LINE$;
11170 PRINT "
          ";LINE$;
11180 PRINT "          ";
11190 RETURN
11200 REM ***********BOAT************
11210 LINE$=""
11220 PRINT "              ";LINE$;
11230 PRINT "     |        ";LINE$;
11240 PRINT "     |
                        ";LINE$;
11250 PRINT "
   ";LINE$;
11260 PRINT "
               ";LINE$;
11270 PRINT "
       ";LINE$;
11280 PRINT "
               ";LINE$;
11290 PRINT "              ";
11295 RETURN
11300 REM ***********TREE************
11310 LINE$=""
11320 PRINT "          ";LINE$;
11330 PRINT "   ^      ";LINE$;
11340 PRINT "
          ";LINE$;
11350 PRINT "
          ";LINE$;
11360 PRINT "
          ";LINE$;
11370 PRINT "
          ";LINE$;
11380 PRINT "          ";LINE$;
11390 PRINT "          ";
11395 RETURN
```

```
10 REM * This program enables the
20 REM * user to create pictures
30 REM * more easily using control
40 REM * graphics characters.
100 PRINT "                         "
110 PRINT "                         "
120 PRINT "                         "
130 PRINT "                         "
140 PRINT "                         "
150 PRINT "                         "
160 PRINT "                         "
170 PRINT "                         "
180 PRINT "                         "
190 PRINT "                         "
200 PRINT "                         "
210 PRINT "                         "
220 PRINT "                         "
230 PRINT "                         "
240 PRINT "                         "
250 PRINT "                         "
260 PRINT "                         "
270 PRINT "                         "
280 PRINT "                         "
290 PRINT "                         "
300 PRINT "                         "
```

```
10 REM SOUND DEMONSTRATION
20 PRINT "}":REM CLEAR SCREEN
30 VOICE=0
40 PRINT "LOW PITCH (0 - 255)";
50 INPUT LOPITCH
60 IF LOPITCH<0 OR LOPITCH>255 THEN 40
70 PRINT "HI PITCH (";LOPITCH;" - 255)";
80 INPUT HIPITCH
90 IF HIPITCH<LOPITCH OR HIPITCH>255 THEN 70
100 PRINT "STARTING VOLUME (0 - 15)";
110 INPUT STARTVOL
120 IF STARTVOL<0 OR STARTVOL>15 THEN 100
130 PRINT "ENDING VOLUME (0 - 15)";
140 INPUT FINISHVOL
150 IF FINISHVOL<0 OR FINISHVOL>15 THEN 130
160 PRINT "DISTORTION (0 - 14)";
170 INPUT DISTORTION
180 IF DISTORTION<0 OR DISTORTION>14 THEN 160
190 FOR PITCH=LOPITCH TO HIPITCH
200 PRINT "PITCH = ";PITCH,"DISTORTION = ";DISTORTION
210 FOR VOLUME=STARTVOL TO FINISHVOL STEP SGN(FINISHVOL-STARTVOL)
220 SOUND VOICE,PITCH,DISTORTION,VOLUME
230 NEXT VOLUME
240 SOUND VOICE,0,0,0:REM TURN OFF SOUND
250 FOR PAUSE=1 TO 20:NEXT PAUSE
260 NEXT PITCH
270 GOTO 20
280 STOP
290 END
```

```
10 REM WORKSHEET:  COLOR MANIPULATION
20 REM MANIPULATES BORDER AND DISPLAY SCREEN COLORS.
30 PRINT "}":REM ESC KEY FOLLOWED BY SHIFT CLEAR.
40 REG=2:REM PLAYFIELD 1
50 GOSUB 200
60 SETCOLOR 2,0,0
70 REG=4:REM BACKGROUND
80 GOSUB 200
90 SETCOLOR 4,0,0
100 GOTO 40
200 FOR HUE=0 TO 15
210 FOR LUM=0 TO 14 STEP 2
220 SETCOLOR REG,HUE,LUM
230 FOR PAUSE=1 TO 30:NEXT PAUSE
240 NEXT LUM
250 NEXT HUE
260 RETURN
270 END
```

```
10 REM WORKSHEET:  COLORED BOX
20 PAUSE=80
30 GRAPHICS 7+16
40 COLOR 1
50 GOSUB 200
60 FOR HUE=0 TO 15
70 FOR LUM=0 TO 14 STEP 2
80 SETCOLOR 0,HUE,LUM
90 GOSUB 300:REM PAUSE
100 NEXT LUM
110 NEXT HUE
120 GOTO 60
200 REM DRAW SQUARE
210 PLOT 90,50
220 DRAWTO 90,30
230 GOSUB 300:REM PAUSE
240 DRAWTO 70,30
250 GOSUB 300:REM PAUSE
260 POSITION 70,50
270 POKE 765,1
280 XIO 18,#6,0,0,"S:"
290 RETURN
300 FOR P=1 TO PAUSE:NEXT P
310 RETURN
320 END
```

```
10 DIM CMD$(1)
100 PRINT "}"
110 PRINT "MODE 3, 5, OR 7";
120 INPUT MODE
130 GRAPHICS MODE+16
200 OPEN #1,4,0,"D:CLOWN.DAT":INPUT #1;GR
220 INPUT #1;CMD$:IF CMD$="D" THEN INPUT #1;X,Y:GOSUB 500:DRAWTO X,Y:GOTO 22
250 IF CMD$="F" THEN INPUT #1;X,Y,Z:GOSUB 500:POSITION X,Y:POKE 765,Z:XIO 18
0,0,"S:":PLOT X,Y:GOTO 220
260 IF CMD$="P" THEN INPUT #1,X,Y:GOSUB 500:PLOT X,Y:GOTO 220
270 IF CMD$="S" THEN INPUT #1,X,Y,Z:SETCOLOR X-1,Y,Z:GOTO 220
280 IF CMD$="C" THEN INPUT #1,X:COLOR X:GOTO 220
300 CLOSE #1
310 GOTO 310
400 END
500 REM SCALING ROUTINE
510 X=X-30
520 IF MODE=7 THEN RETURN
530 IF MODE=5 THEN X=INT(X/2):Y=INT(Y/2)
540 IF MODE=3 THEN X=INT(X/4):Y=INT(Y/4)
550 RETURN
560 END
```

```
10 REM CHARACTER GRAPHICS
20 PRINT "}"
30 PRINT "GRAPHICS 1 OR 2";
40 INPUT G
50 IF G<>1 AND G<>2 THEN 30
60 GRAPHICS G+16
70 POSITION 5,3
80 PRINT #6;"N":REM UPPER CASE N
90 POSITION 6,4
100 PRINT #6;"e":REM LOWER CASE E
110 POSITION 7,5
120 PRINT #6;"A":REM UPPER CASE INVERSE VIDEO A
130 POSITION 8,6
140 PRINT #6;"t":REM LOWER CASE INVERSE VIDEO T
150 FOR COL=0 TO 3
160 HUE=INT(16*RND(0))
170 FOR LUM=0 TO 14 STEP 2
180 SETCOLOR COL,HUE,LUM
190 FOR PAUSE=1 TO 40:NEXT PAUSE
200 NEXT LUM
210 SETCOLOR COL,HUE,8
220 NEXT COL
230 GOTO 150
240 END
```

```
10 REM *  ANIMATED ARROW  *
20 GRAPHICS 7+16
30 Y=40
40 FOR HUE=0 TO 15
50 FOR X=15 TO 105 STEP 5
60 SETCOLOR 0,HUE,2*X/15
70 REM *  DRAW ARROW  *
80 COLOR 1
90 GOSUB 10800
100 REM *  ERASE ARROW  *
110 COLOR 0
120 GOSUB 10800
130 NEXT X
140 NEXT HUE
150 END
10800 REM ******Makes Arrow*********
10810 PLOT X,Y
10815 DRAWTO X-15,Y
10820 DRAWTO X-15,Y+2
10830 DRAWTO X,Y+2
10840 PLOT X+2,Y+1
10850 DRAWTO X-6,Y-3
10860 PLOT X+2,Y+1
10870 DRAWTO X-6,Y+5
10880 RETURN
```

```
10 REM COLORED CUBE
20 PRINT "}":OPEN #1,4,0,"K:"
30 PRINT "YOU CAN CHANGE THE COLORS OF"
40 PRINT "THE CUBE FACES BY HITTING DIFFERENT"
50 PRINT "KEYS ON THE KEYBOARD."
60 PRINT
70 PRINT "THE CUBE ONLY LOOKS REASONABLE IN"
80 PRINT "GRAPHICS MODES 3, 5, OR 7, BUT"
90 PRINT "YOU CAN TRY OTHER MODES."
100 PRINT "TYPE THE SPACE BAR WHEN YOU WANT"
110 PRINT "TO TRY A DIFFERENT MODE."
120 PRINT
130 PRINT "GRAPHICS MODE";
140 INPUT G:GRAPHICS G+16
150 FOR I=0 TO 3:SETCOLOR I,0,14:NEXT I:SETCOLOR 4,9,4
160 X=12:Y=9
170 COLOR 1
180 FOR I=0 TO 10
190 PLOT X,Y+I:DRAWTO X+10,Y+I
200 NEXT I
210 COLOR 2
220 FOR I=1 TO 6
230 PLOT X+I,Y-I:DRAWTO X+I+10,Y-I
240 NEXT I
250 COLOR 3
260 FOR I=1 TO 6
270 PLOT X+10+I,Y-I:DRAWTO X+10+I,Y+10-I
280 NEXT I
290 FOR I=0 TO 2
300 GET #1,KEY
310 IF KEY=32 THEN PRINT "}":GOTO 130
320 IF KEY<48 THEN KEY=48
330 SETCOLOR I,1,2*(KEY-48)
340 NEXT I
350 GOTO 290
360 STOP
370 END
```

APPENDIX B
ERROR MESSAGES

ERROR CODES

ERROR CODE	ERROR CODE MESSAGE
2	Memory Insufficient
3	Value Error
4	Too Many Variables
5	String Length Error
6	Out of Data Error
7	Number greater than 32767
8	Input Statement Error
9	Array or String DIM Error
10	Argument Stack Overflow
11	Floating Point Overflow/ Underflow Error
12	Line Not Found
13	No Matching FOR Statement
14	Line Too Long Error
15	GOSUB or FOR Line Deleted
16	RETURN Error
17	Garbage Error
18	Invalid String Character

Note: The following are INPUT/OUTPUT errors that result during the use of disk drives, printers, or other accessory devices. Further information is provided with the auxiliary hardware.

ERROR CODE	ERROR CODE MESSAGE
19	LOAD program Too Long
20	Device Number Larger
21	LOAD File Error
128	BREAK Abort
129	IOCB
130	Nonexistent Device
131	IOCB Write Only
132	Invalid Command
133	Device or File not Open
134	BAD IOCB Number
135	IOCB Read Only Error
136	EOF
137	Truncated Record
138	Device Timeout
139	Device NAK
140	Serial Bus
141	Cursor Out of Range
142	Serial Bus Data Frame Overrun
143	Serial bus data frame checksum error
144	Device done error
145	Read after write compare error
146	Function not implemented
147	Insufficient RAM
160	Drive number error
161	Too many OPEN files
162	Disk full
163	Unrecoverable system data I/O error
164	File number mismatch
165	File name error
166	POINT data length error
167	File locked
168	Command invalid
169	Directory full
170	File not found
171	POINT invalid

ERROR MESSAGES

ERROR CODE NO.	ERROR CODE MESSAGE
2	**Memory insufficient** to store the statement or the new variable name or to DIM a new string variable.
3	**Value Error:** A value expected to be a positive integer is negative, a value expected to be within a specific range is not.
4	**Too Many Variables:** A maximum of 128 different variable names is allowed. (See **Variable Name Limit**.)
5	**String Length Error:** Attempted to store beyond the DIMensioned string length.
6	**Out of Data Error:** READ statement requires more data items than supplied by DATA statement(s).
7	**Number greater than 32767:** Value is not a positive integer or is greater than 32767.
8	**Input Statement Error:** Attempted to INPUT a non-numeric value into a numeric variable.
9	**Array or String DIM Error:** DIM size is greater than 32767 or an array/martix reference is out of the range of the dimensioned size, or the array/matrix or string has been already DIMensioned, or a reference has been made to an undimensioned array or string.
10	**Argument Stack Overflow:** There are too many GOSUBs or too large an expression.
11	**Floating Point Overflow/Underflow Error:** Attempted to divide by zero or refer to a number larger than 1×10^{98} or smaller than 1×10^{-99}.
12	**Line Not Found:** A GOSUB, GOTO, or THEN referenced a non-existent line number.
13	**No Matching FOR Statement:** A NEXT was encountered without a previous FOR, or nested FOR/NEXT statements do not match properly. (Error is reported at the NEXT statement, not at FOR).
14	**Line Too Long Error:** The statement is too complex or too long for BASIC to handle.
15	**GOSUB or FOR Line Deleted:** A NEXT or RETURN statement was encountered and the corresponding FOR or GOSUB has been deleted since the last RUN.

ERROR CODE NO.	ERROR CODE MESSAGE
16	**RETURN Error:** A RETURN was encountered without a matching GOSUB.
17	**Garbage Error:** Execution of "garbage" (bad RAM bits) was attempted. This error code may indicate a hardware problem, but may also be the result of faulty use of POKE. Try typing NEW or powering down, then re-enter the program without any POKE commands.
18	**Invalid String Character:** String does not start with a valid character, or string in VAL statement is not a numeric string.
Note:	**The following are INPUT/OUTPUT errors that result during the use of disk drives, printers, or other accessory devices. Further information is provided with the auxiliary hardware.**
19	**LOAD program Too Long:** Insufficient memory remains to complete LOAD.
20	**Device Number Larger** than 7 or Equal to 0.
21	**LOAD File Error:** Attempted to LOAD a non-LOAD file.
128	**BREAK Abort:** User hit BREAK key during I/O operation.
129	**IOCB**[1] already open.
130	**Nonexistent Device** specified.
131	**IOCB Write Only.** READ command to a write-only device (Printer).
132	**Invalid Command:** The command is invalid for this device.
133	**Device or File not Open:** No OPEN specified for the device.
134	**Bad IOCB Number:** Illegal device number.
135	**IOCB Read Only Error:** WRITE command to a read-only device.
136	**EOF:** End of File read has been reached. (***NOTE:*** This message may occur when using cassette files.)
137	**Truncated Record:** Attempt to read a record longer than 256 characters.
138	**Device Timeout.** Device doesn't respond.
139	**Device NAK:** Garbage at serial port or bad disk drive.
140	**Serial bus** input framing error.
141	**Cursor out of range** for particular mode.
142	**Serial bus data frame overrun.**

[1]IOCB refers to Input/Output Control Block. The device number is the same as the IOCB number.

ERROR CODE NO.	ERROR CODE MESSAGE
143	**Serial bus data frame checksum error.**
144	**Device done error** (invalid "done" byte): Attempt to write on a write-protected diskette.
145	**Read after write compare error** (disk handler) or bad screen mode handler.
146	**Function not implemented** in handler.
147	**Insufficient RAM** for operating selected graphics mode.
160	**Drive number error.**
161	**Too many OPEN files** (no sector buffer available).
162	**Disk full** (no free sectors).
163	**Unrecoverable system data I/O error.**
164	**File number mismatch:** Links on disk are messed up.
165	**File name error.**
166	**POINT data length error.**
167	**File locked.**
168	**Command invalid** (special operation code).
169	**Directory full** (64 files).
170	**File not found.**
171	**POINT invalid.**

APPENDIX C
BASIC RESERVED WORDS

ALPHABETICAL DIRECTORY OF BASIC RESERVED WORDS

Note: The period is mandatory after all abbreviated keywords.

RESERVED WORD:	ABBREVIATION:	BRIEF SUMMARY OF BASIC STATEMENT
ABS		Function returns absolute value (unsigned) of the variable or expression.
ADR		Function returns memory address of a string.
AND		Logical operator: Expression is true only if both subexpressions joined by **AND** are true.
ASC		String function returns the numeric value of a single string character.
ATN		Function returns the arctangent of a number or expression in radians or degrees.
BYE	B.	Exit from BASIC and return to the resident operating system or console processor.
CLOAD	CLOA.	Loads data from Program Recorder into RAM.
CHR$		String function returns a single string byte equivalent to a numeric value between 0 and 255 in ATASCII code.
CLOG		Function returns the base 10 logarithm of an expression.
CLOSE	CL.	I/O statement used to close a file at the conclusion of I/O operations.
CLR		The opposite of DIM: Undimensions all strings; matrices.
COLOR	C.	Chooses color register to be used in color graphics work.
COM		Same as DIM.
CONT	CON.	Continue. Causes a program to restart execution on the next line following use of the BREAK key or encountering a **STOP**.
COS		Function returns the cosine of the variable or expression (degrees or radians).
CSAVE		Outputs data from RAM to the Program Recorder for tape storage.

RESERVED WORD:	ABBREVIATION:	BRIEF SUMMARY OF BASIC STATEMENT
DATA	D.	Part of **READ/DATA** combination. Used to identify the succeeding items (which must be separated by commas) as individual data items.
DEG	DE.	Statement **DEG** tells computer to perform trigonometric functions in degrees instead of radians. (Default in radians.)
DIM	DI.	Reserves the specified amount of memory for matrix, array, or string. All string variables, arrays, matrices must be dimensioned with a DIM statement.
DOS	DO.	Reserved word for disk operators. Causes the menu to be displayed. (See *DOS Manual*.)
DRAWTO	DR.	Draws a straight line between a plotted point and specified point.
END		Stops program execution; closes files; turns off sounds. Program may be restarted using **CONT**. (Note: **END** may be used more than once in a program.)
ENTER	E.	I/O command used to store data or programs in untokenized (source) form.
EXP		Function returns e (2.7182818) raised to the specified power.
FOR	F.	Used with **NEXT** to establish **FOR/NEXT** loops. Introduces the range that the loop variable will operate in during the execution of loop.
FRE		Function returns the amount of remaining user memory (in bytes).
GET	GE.	Used mostly with disk operations to input a single byte of data.
GOSUB	GOS.	Branch to a subroutine beginning at the specified line number.
GOTO	G.	Unconditional branch to a specified line number.
GRAPHICS	GR.	Specifies which of the eight graphics modes is to be used. **GR.0** may be used to clear screen.
IF		Used to cause conditional branching or to execute another statement on the same line (only if the first expression is true).
INPUT	I.	Causes computer to ask for input from keyboard. Execution continues only when RETURN key is pressed after inputting data.
INT		Function returns the next lowest whole integer below the specified value. Rounding is always downward, even when number is negative.
LEN		String function returns the length of the specified string in bytes or characters (1 byte contains 1 character).

RESERVED WORD:	ABBREVIATION:	BRIEF SUMMARY OF BASIC STATEMENT
LET	LE.	Assigns a value to a specific variable name. LET is optional in Atari BASIC, and may be simply omitted.
LIST	L.	Display or otherwise output the program list.
LOAD	LO.	Input from disk, etc. into the computer.
LOCATE	LOC.	Graphics: Stores, in a specified variable, the value that controls a specified graphics point.
LOG		Function returns the natural logarithm of a number.
LPRINT	LP.	Command to line printer to print the specified message.
NEW		Erases all contents of user RAM.
NEXT	N.	Causes a **FOR/NEXT** loop to terminate or continue depending on the particular variables or expressions. All loops are executed at least once.
NOT		A "1" is returned only if the expression is NOT true. If it is true, a "0" is returned.
NOTE	NO.	See *DOS/FMS Manual*...used only in disk operations.
ON		Used with **GOTO** or **GOSUB** for branching purposes. Multiple branches to different line numbers are possible depending on the value of the **ON** variable or expression.
OPEN	O.	Opens the specified file for input of output operations.
OR		Logical operator used between two expressions. If either one is true, a "1" is evaluated. A "0" results only if both are false.
PADDLE		Function returns position of the paddle game controller.
PEEK		Function returns decimal form of contents of specified memory location (RAM or ROM).
PLOT	PL.	Causes a single point to be plotted at the X,Y location specified.
POINT	P.	Used with disk operations only.
POKE	POK.	Insert the specified byte into the specified memory location. May be used only with RAM. Don't try to POKE ROM or you'll get an error.
POP		Removes the loop variable from the **GOSUB** stack. Used when departure from the loop is made in other than normal manner.
POSITION	POS.	Sets the cursor to the specified screen position.
PRINT	PR. or ?	I/O command causes output from the computer to the specified output device.

RESERVED WORD:	ABBREVIATION:	BRIEF SUMMARY OF BASIC STATEMENT
PTRIG		Function returns status of the trigger button on game controllers.
PUT	**PU.**	Causes output of a single byte of data from the computer to the specified device.
RAD		Specifies that information is in radians rather than degrees when using the trigonometric functions. Default is to **RAD**. (See **DEG**.)
READ	**REA.**	Read the next items in the **DATA** list and assign to specified variables.
REM	**R.** or **.** SPACE	Remarks. This statement does nothing, but comments may be printed within the program list for future reference by the programmer. Statements on a line that starts with **REM** are not executed.
RESTORE	**RES.**	Allows **DATA** to be **read** more than once.
RETURN	**RET.**	**RETURN** from subroutine to the statement immediately following the one in which **GOSUB** appeared.
RND		Function returns a random number between 0 and 1, but never 1.
RUN	**RU.**	Execute the program. Sets normal variables to 0, undims arrays and string.
SAVE	**S.**	I/O statement causes data or program to be recorded on disk under filespec provided with **SAVE**.
SETCOLOR	**SE.**	Store hue and luminance color data in a particular color register.
SGN		Function returns +1 if value is positive, 0 if zero, -1 if negative.
SIN		Function returns trigonometric sine of given value (**DEG** or **RAD**).
SOUND	**SO.**	Controls register, sound pitch, distortion, and volume of a tone or note.
SQR		Function returns the square root of the specified value.
STATUS	**ST.**	Calls status routine for specified device.
STEP		Used with **FOR/NEXT**. Determines quality to be skipped between each pair of loop variable values.
STICK		Function returns position of stick game controller.
STRIG		Function returns 1 if stick trigger button not pressed, 0 if pressed.
STOP	**STO.**	Causes execution to stop, but does not close files or turn off sounds.

RESERVED WORD:	ABBREVIATION:	BRIEF SUMMARY OF BASIC STATEMENT
STR$		Function returns a character string equal to numeric value given. For example: **STR$(65)** returns 65 as a string.
THEN		Used with **IF**: If expression is true, the **THEN** statements are executed. If the expression is false, control passes to next line.
TO		Used with **FOR** as in "FOR X = 1 TO 10". Separates the loop range expressions.
TRAP	**T.**	Takes control of program in case of an **INPUT** error and directs execution to a specified line number.
USR		Function returns results of a machine-language subroutine.
VAL		Function returns the equivalent numeric value of a string.
XIO	**X.**	General I/O statement used with disk operations (see *DOS/FMS Manual*) and in graphics work (Fill).

APPENDIX D

ATASCII CHARACTER SET

ATASCII CHARACTER SET

DECIMAL CODE	HEXADECIMAL CODE	CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	CHARACTER
0	0	♥	13	D	▔	26	1A	└
1	1	├	14	E	▁	27	1B	␛
2	2	▕	15	F	▖	28	1C	↑
3	3	┘	16	10	♣	29	1D	↓
4	4	┤	17	11	┌	30	1E	←
5	5	┐	18	12	─	31	1F	→
6	6	╱	19	13	┼	32	20	Space
7	7	╲	20	14	●	33	21	!
8	8	◢	21	15	▄	34	22	"
9	9	▗	22	16	▎	35	23	#
10	A	◣	23	17	┬	36	24	$
11	B	▝	24	18	┴	37	25	%
12	C	▘	25	19	▌	38	26	&

DECIMAL CODE	HEXADECIMAL CODE	CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	CHARACTER
39	27	'	55	37	7	71	47	G
40	28	(	56	38	8	72	48	H
41	29	)	57	39	9	73	49	I
42	2A	*	58	3A	:	74	4A	J
43	2B	+	59	3B	;	75	4B	K
44	2C	,	60	3C	<	76	4C	L
45	2D	-	61	3D	=	77	4D	M
46	2E	.	62	3E	>	78	4E	N
47	2F	/	63	3F	?	79	4F	O
48	30	0	64	40	@	80	50	P
49	31	1	65	41	A	81	51	Q
50	32	2	66	42	B	82	52	R
51	33	3	67	43	C	83	53	S
52	34	4	68	44	D	84	54	T
53	35	5	69	45	E	85	55	U
54	36	6	70	46	F	86	56	V

DECIMAL CODE	HEXADECIMAL CODE	CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	CHARACTER
87	57	W	103	67	g	119	77	w
88	58	X	104	68	h	120	78	x
89	59	Y	105	69	i	121	79	y
90	5A	Z	106	6A	j	122	7A	z
91	5B	[	107	6B	k	123	7B	♠
92	5C	\	108	6C	l	124	7C	|
93	5D	]	109	6D	m	125	7D	↰
94	5E	^	110	6E	n	126	7E	◀
95	5F	_	111	6F	o	127	7F	▶
96	60	◆	112	70	p	128	80	
97	61	a	113	71	q	129	81	
98	62	b	114	72	r	130	82	
99	63	c	115	73	s	131	83	
100	64	d	116	74	t	132	84	
101	65	e	117	75	u	133	85	
102	66	f	118	76	v	134	86	

DECIMAL CODE	HEXADECIMAL CODE	CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	CHARACTER
135	87		151	97		167	A7	
136	88		152	98		168	A8	
137	89		153	99		169	A9	
138	8A		154	9A		170	AA	
139	8B		155	9B	(EOL) RETURN	171	AB	
140	8C		156	9C	↑	172	AC	
141	8D		157	9D	↓	173	AD	
142	8E		158	9E	←	174	AE	
143	8F		159	9F	→	175	AF	
144	90		160	A0		176	B0	
145	91		161	A1		177	B1	
146	92		162	A2		178	B2	
147	93		163	A3		179	B3	
148	94		164	A4		180	B4	
149	95		165	A5		181	B5	
150	96		166	A6		182	B6	